

KNOWLEDGE VISUALIZATION: FROM THEORY TO PRACTICE

by

Dong Hyun Jeong

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Information Technology

Charlotte

2010

Approved by:

Dr. William Ribarsky

Dr. Larry F. Hodges

Dr. Zachary Wartell

Dr. Jing Yang

Dr. Andrew Willis

©2010
Dong Hyun Jeong
ALL RIGHTS RESERVED

ABSTRACT

DONG HYUN JEONG. Knowledge visualization: from theory to practice. (Under the direction of DR. WILLIAM RIBARSKY)

Visualizations have been known as efficient tools that can help users analyze complex data. However, understanding the displayed data and finding underlying knowledge is still difficult. In this work, a new approach is proposed based on understanding the definition of knowledge. Although there are many definitions used in different areas, this work focuses on representing knowledge as a part of a visualization and showing the benefit of adopting knowledge representation. Specifically, this work begins with understanding interaction and reasoning in visual analytics systems, then a new definition of knowledge visualization and its underlying knowledge conversion processes are proposed. The definition of knowledge is differentiated as either explicit or tacit knowledge. Instead of directly representing data, the value of the explicit knowledge associated with the data is determined based on a cost/benefit analysis. In accordance to its importance, the knowledge is displayed to help the user understand the complex data through visual analytical reasoning and discovery.

ACKNOWLEDGMENTS

Throughout my time at UNC Charlotte, I have had the great fortune to work with outstanding colleagues. Therefore, I dedicate this thesis to the colleagues who have directly and indirectly contributed to this thesis.

I would like to thank to my advisor, William Ribarsky, who has provided everything I needed to succeed and repeatedly put my efforts in building a new research area called knowledge visualization as my thesis topic. I also thank to my committee members, Larry F. Hodges, Zachary Wartell, Jing Yang, and Andrew Willis for their feedback and comments.

As a student member in the Charlotte Visualization Center, I have been involved into several projects and continuously communicate with people to share ideas and research skills. I gracefully thank to people, Remco Chang, Tom Butkiewicz, Caroline Ziemkiewicz, Xiaoyu Wang, Wenwen Dou, and Tera Green. During my PhD study, Remco gave me a lot of helpful comments and research ideas. Without his help, I would have not been successful in formalizing my ideas turn into research papers.

In looking back at how I came to Charlotte, I must thank Larry F. Hodges for inviting me as a visiting researcher about six years ago. As a visiting researcher, I have learned a lot about how to build communication and collaboration skills. Without him, I would have not done my PhD journey here at UNC Charlotte successfully.

Lastly, I want to thank my parents, Jae-won Jeong and Jeong-ja Um, and my brother, Su-hyun Jeong. They have always provided mental and financial supports for me to create a flexible, but internally concrete life. Also I want to thank my girl

friend, Soo-Yeon Ji. She has always encouraged me to have a creative thinking on designing a visualization application. From the beginning of my PhD study, she was my core partner and provide innovative ideas and feedbacks related to my research field.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION	1
CHAPTER 2: DEFINITION OF KNOWLEDGE	4
2.1 Philosophical Definition of Knowledge	4
2.2 Scientific Definition of Knowledge	5
2.3 Properties of Knowledge	8
CHAPTER 3: KNOWLEDGE VISUALIZATION FRAMEWORK	14
3.1 Knowledge Visualization Model	15
3.2 Maximizing the Value of Visualization	18
CHAPTER 4: KNOWLEDGE CONVERSION PROCESSES	27
4.1 Internalization	27
4.1.1 Interactive Visual Analysis of Time-series Microarray Data	31
4.1.1.1 Related Work	33
4.1.1.2 A Dynamic Model	37
4.1.1.3 Designing a Visualization Model	39
4.1.1.4 Visual Representation	41
4.1.1.5 Pixel-based Gene Representation	45
4.1.1.6 Data Positioning in 2D View	48
4.1.1.7 Visual Analysis	51
4.1.1.8 Building up the Biological Insights	57
4.1.1.9 Discussion	60

4.1.2	iPCA: An Interactive System for PCA-based Visual Analytics	61
4.1.2.1	Related Work	63
4.1.2.2	Interface Design	66
4.1.2.3	Interaction	68
4.1.2.4	Evaluation	72
4.1.2.5	Discussion	79
4.2	Externalization	80
4.2.1	Understanding experts' reasoning process from interactions	83
4.2.1.1	Related Work	84
4.2.1.2	WireVis Interactions	87
4.2.1.3	Evaluation	96
4.2.1.4	Results	101
4.2.1.5	Discussion	107
4.2.2	Voting Viz: Perception of agreement or disagreement of two or more ideas	110
4.2.2.1	Interface Design	111
4.2.2.2	Voting and Preserving the Users' Reasoning	116
4.2.2.3	Interactions	119
4.2.2.4	Discussion	122
4.3	Combination	124
4.3.1	Adopting Knowledge in Visualization	127
4.3.1.1	Examples	128
4.3.1.2	Discussion	131

	viii
4.4 Collaboration	131
4.4.1 A Continuous Analysis Process Between Desktop And Collaborative Visual Analytics Environments	135
4.4.1.1 Related Work	136
4.4.1.2 Converting a VA Tool From Desktop to Tabletop	138
4.4.1.3 Sharing Analytical Processes	143
4.4.1.4 Challenges	148
4.4.1.5 Discussion	151
4.4.2 Understanding Users' Knowledge Sharing Process	153
4.4.2.1 Understanding Collaborative Analysis Procedures	153
4.4.2.2 Conducting a Study	156
4.4.2.3 Discussion	162
CHAPTER 5: DESIGNING KNOWLEDGE VISUALIZATION	164
5.1 Design Issues	165
5.2 Genomic Visualization (GVis)	167
5.2.1 GVis Framework	168
5.2.1.1 Representing the Phylogeny Structure	169
5.2.1.2 Interactions in GVis	171
5.2.1.3 Performing Sequence Analysis	173
5.3 Knowledge Visualization	174
5.3.1 Creating Knowledge Structure Topology	175
5.3.2 Labeling and Annotation	178
5.3.2.1 Applying the Cost/ Benefit Analysis	178

5.3.2.2	Bringing Out the Information from the Bottom to the Top	180
5.3.3	Interaction Supports	184
5.3.3.1	Knowledge Discovering	184
5.3.3.2	Knowledge Building	185
5.3.4	Evaluation	186
5.3.4.1	Procedure	187
5.3.4.2	Evaluation Results	189
5.3.4.3	Discussion	191
CHAPTER 6:	CONCLUSION AND FUTURE WORK	192
REFERENCES		194

CHAPTER 1: INTRODUCTION

Visualization is viewed as an important tool with which users can increase the ability of exploring and discovering new patterns or relations in large-scale data. Because of this reason, there have been many research domains, such as bioinformatics, chemistry, etc., that have adopted visualization techniques to maximize understanding of data or to unveil hidden but important knowledge. However, for resolving the problems that exist in different domains, a single visualization or visual analysis is insufficient. There is rather the need for an analytical reasoning process where interactive visualization is an integral part. To address this need, the research agenda by Thomas and Cook [136] outlines the fundamental research areas for the field of visual analytics based on the considerations of data representations, interactive techniques, perceptual effects, etc. However, there is a need to go beyond typical considerations of visualization in terms of data representations, interactive techniques, and perceptual effects. What is needed is a visualization approach in terms of knowledge products that are closer to the reasoning artifacts needed in complex analytical processes. The purpose of this thesis is to identify in more detail what is needed and to suggest an initial path and structure to provide it. Our hypothesis is that since the goal of visual analysis is in acquiring more knowledge, an approach that emphasizes the knowledge products could be more direct and beneficial. We call this approach “knowledge vi-

sualization.” In this thesis, we present the initial path and focus on identifying what knowledge visualization is and what is needed to make it possible.

What is “knowledge visualization”? This will become clearer in subsequent sections of this thesis, but, to begin with, it has to do with dealing with knowledge representations rather than data representations and then determining what is the highest value knowledge representation for a given task. There are several difficulties in doing this. The task, for one thing, is not usually well defined, at least initially. For the complex reasoning processes discussed above, the user is exploring and attempting to “discover the unexpected”, as stated in the visual analytics mission statement [136]. For another thing, the task can change, it can indeed evolve, which could cause the knowledge visualization to change significantly and perhaps dynamically. (It will be seen that the change is not just in terms of deciding what to display or not display, but can be in terms of the visual representation itself.) We will address these issues with examples and a concrete implementation for knowledge visualization.

In recent years, visualization experts have begun to think about adopting knowledge in visualization [19, 27]. We caution the reader that the term “knowledge visualization” has been used before in different contexts [19, 45, 75, 126]. However, most of these uses are fuzzy (some not even distinguishing clearly between information and knowledge) and are all highly conceptual with no detailed vision of how to model let alone implement knowledge visualizations and no clear idea about how they are distinguished from other forms of visualization. Since we are starting fresh and making a significant initial effort to do these things, we feel justified in taking the term knowledge visualization for our own purposes. It is too good a term to leave to these

other, unfocused efforts. This thesis has the following significant outcomes.

- It starts with a definition of knowledge and its properties and shows how some basic, general design considerations that emanate from these.
- It provides examples and study results on how people build and share knowledge within visualization.
- It shows how a knowledge visualization framework can be built from this definition, properties, and design considerations, and how one can calculate the value of visualizations within this framework.
- It demonstrates how the knowledge framework can be used in a specific application resulting in improved support of discovery, insight-building, and reasoning.
- It represents the usefulness of the knowledge visualization application by conducting an evaluation study.

CHAPTER 2: DEFINITION OF KNOWLEDGE

As we mentioned in the previous chapter, to make a concrete knowledge visualization framework, it is important to begin with understanding the definition of knowledge. There are many definitions, of course, but we start with understanding the existing definitions of knowledge into two categories: philosophical and scientific.

2.1 Philosophical Definition of Knowledge

The history of philosophy since the ancient Greek period has focused on seeking the answer for the question, “what is knowledge?” This branch of philosophy that studies the nature of knowledge is called **epistemology**. In epistemology, there are two opposing traditions as rationalism and empiricism. Rationalism argues that knowledge can be obtained deductively by reasoning. In contrast, empiricism claims that knowledge can be attained inductively from sensory experiences and no prior knowledge is required. Although there are fundamental differences between rationalism and empiricism, philosophers generally agreed that knowledge is “justified true belief.” In the 18th century, Kant believes that knowledge arises when both the logical thinking of rationalism and sensory experience of empiricism work together. This philosophical understanding of knowledge is continuously stretched to form scientific definition of knowledge. Machlup [85] classified several types of knowledge based on his consideration of “how knowledge can be ‘true,’ tested, and verified.” His work clarifies the

distinctions between information, knowledge, and data, and has been broadly applied to form the scientific definition of knowledge [146].

2.2 Scientific Definition of Knowledge

In knowledge management literature, the distinction between data, information, and knowledge is believed to be important when designing knowledge management programs [76]. Although it is not clearly known when and by whom the relationships between data, information, knowledge and wisdom were presented as a DIKW hierarchy, the DIKW has been a part of the language of information science for many years. Based on understanding the relationships, Zeleny [159] proposes the DIKW pyramid that presents data as the broad base of the pyramid and wisdom as its much smaller summit. Ackoff [4] defines an additional component - “understanding”, to the hierarchy, placing it between wisdom and knowledge. He claims that knowledge become wisdom only through a human’s understanding process. By reviewing various definitions and explanations of DIKW hierarchy, Syed and Shah [128] creates a model to explicate the relationship between data, information, knowledge, and wisdom. Although many researchers have tried to find a clear definition of knowledge, most people claimed that many of the definitions are neither concise nor precise and the definition is relative and context dependent [128].

However, one of the most commonly quoted definitions is from Davenport and Prusak [42]. They state that “knowledge derives from information as information derives from data” and further define knowledge as “a fluid mix of framed experience, contextual information, values and expert insight that provides a framework for eval-

uating and incorporating new experiences and information.” From Davenport and Prusak’s perspective, knowledge is the refined information in which human cognition has added value. In other words, information becomes knowledge through cognitive effort. Based on their definition, we think that knowledge can only result from human cognitive process that includes perceiving, recognizing, conceiving judging, reasoning, and imagining [1]. It also shows that knowledge fundamentally involves relationships either among information or other pieces of knowledge.

Prior to that, Polanyi [107] creates a concept of tacit knowledge and further explains the difference between tacit knowing and tacit knowledge. He specifically focuses on perception that determines and shapes knowledge, and defines the tacit knowledge that is valuable because it provides contexts for people, places, ideas, and experiences. He believes that tacit knowledge can be built through a tacit knowing process, and transferred between people. His concern falls into understanding the tacit knowing is not just perceptually looking at an object, but also understanding the underlying meaning from the object. It is somewhat related to combining the tacit knowing process with other forms of perception and awareness. Since his primary interest is in the process of scientific discovery, he believes that there is a clear distinction between tacit knowledge and explicit knowledge. He specifically describes that “while tacit knowledge can be processed by itself, explicit knowledge must rely on being tacitly understood and applied.” Examples of explicit knowledge such as a document, a recording, or a work of art can be meaningful only through the active process of tacit knowing. However, his distinction between tacit and explicit knowledge has not been broadly agreed by other philosophers because he did not provide sufficient

justification on each term. Polanyi [107] believes that every knowledge is performed personally through a personalized tacit knowing process. However, Popper [108] found that Polanyi's definition is not applicable to other types of knowledge and claims that "scientific knowledge is independent of the mind of any particular individual and truly exists in and belongs to the objective process of knowing." Later, Crowley [37] defines the definition of tacit knowledge as "tacit knowledge is highly personal and hard to formalize, making it difficult to communicate or to share with others." The definition is completely different compared to the Polanyi's understanding on tacit knowledge.

In 1995, Nonaka and Takeuchi [97] adopt Polanyi's definition of tacit knowledge to understand how knowledge is shaped and how knowledge can be applied. Specifically, they define two terms of knowledge: tacit and explicit. Tacit knowledge indicates personalized knowledge and explicit knowledge represents formalized knowledge. They divide tacit knowledge into two dimensions - technical and cognitive. The technical dimension encapsulates the meaning of "know-how," and the cognitive dimension "consists of schemata, mental models, beliefs, and perceptions so ingrained that we take them for granted." Since tacit knowledge involves human cognitive understanding, it can only be extracted by a human. However, explicit knowledge can be processed by a computer, transmitted electronically, or stored in a database. Based on understanding tacit and explicit knowledge, they define four knowledge conversion processes: internalization, externalization, socialization, and combination. Internalization is the process of understanding explicit knowledge to form tacit knowledge by individuals. Externalization is considered to form explicit knowledge from tacit knowledge. Socialization is to exchange tacit knowledge among members to create

common mental models and abilities. And combination is the process of combining or reconfiguring bodies of existing explicit knowledge to generate new explicit knowledge. In this thesis, we use Nonaka and Takeuchi’s two terms of knowledge to form a knowledge visualization framework. Prior to designing the framework, we uses the four knowledge conversion processes to understand how knowledge can be converted in visualization and to build the knowledge visualization framework. A detailed explanation how both tacit and explicit knowledge are applied to the knowledge visualization framework is given in Chapter 3.

2.3 Properties of Knowledge

As we explained above, there are many similar and different definitions of knowledge. But, we shall use a restricted definition that supports an implementation supporting the analytical reasoning and discovery that we pursue here. In this view, knowledge, as defined by John Locke [21], is the “perception of agreement or disagreement of two ideas.” This definition indicates that knowledge is “a fluid mix of framed experience, contextual information, values and expert insight that provides a framework for evaluating and incorporating new experiences and information” [42] and taking action [46]. From theses definitions, we understand that knowledge can only result from human cognitive processing (that is, reasoning); raw, undigested data does not qualify. It also is observed that knowledge fundamentally involves relationships, among ideas, processes, or other pieces of knowledge. Finally, knowledge is contextual and is developed with a point of view and, perhaps, with a model or process in mind. Thus, a principal use of knowledge is to build new knowledge and often

to provide a basis for taking action (although the action may be tacit and mostly intellectual, i.e., to understand).

One of the advantages of thinking about the knowledge visualization framework is that we can consider and incorporate general properties of knowledge, which can influence the way we approach knowledge visualizations, where we might expect pay-offs or benefits, and even some general design choices we might make. We can do this even before identifying a particular, detailed approach. With this in mind, we specify the following general properties:

1. Knowledge is of higher value than information or data.
2. Knowledge begets knowledge.
3. Knowledge is compact.
4. Knowledge is connected (more connections, more value).
5. Labeling is important (also, captions, titles, text annotations).
6. Knowledge artifacts are the elements of reasoning.
7. Some knowledge is or can be made independent of user and context (including domain).

Most of these properties are self-evident and require only a structure and process to apply effectively. A couple of properties (knowledge is compact; labeling is important) may need additional explanations (see the following) to sharpen the argument about

their utility. In the following paragraphs, we will describe a structure and process that provides an environment for embedding these properties.

To demonstrate that knowledge is compact, we present a real life example relating to weather data and analysis. The National Weather Service uses Doppler Radar¹ to track detailed weather patterns, including severe storms and tornadoes [96]. The radar volume is collected in a sequence of conical scans at increasing angle every 5-7 minutes. There are over 140 overlapping radars in the continental U.S. Researchers at the Cooperative Institute for Mesoscale Meteorological Studies (CIMMS) at the University of Oklahoma have developed phenomenological analyses of the weather that the Doppler radars reveal [135]. These analyses are completed in real-time as the Doppler data are collected, and the products accompany the complete radar volumes sent to meteorologists at the National Weather Service or to researchers for further analysis. The analyses find mesocyclones (high shear cells), severe storm cells (local areas of heavy precipitation), and even tornado signatures. As shown in Figure 1, these track very well with storm fronts. Generally, an accumulation of mesocyclones and severe storm cells show where significant storm activity is, and certain patterns indicate the conditions for damaging storms or tornadoes. Further, the cells have predictive capability in that their paths can be reasonably projected for an hour or so. But these analysis products are of the order of 10s of KB, a factor of 1000 or more smaller than the original Doppler volume. Thus these analysis products, the concentrated outcomes of human reasoning and expertise about weather phenomena,

¹Doppler radar is a method to produce data by beaming a microwave signal towards a desired target object and listening for its reflection (i.e. doppler effect).

can be used to make decisions (which radar volumes to look at more closely or to save), form knowledge-based searches (for selected regions and time ranges, when are there severe storms of certain types or that may possibly contain tornadoes), and gain understanding (how do storm phenomena relate to each other under different conditions or relate to topographic or man-made features over time).

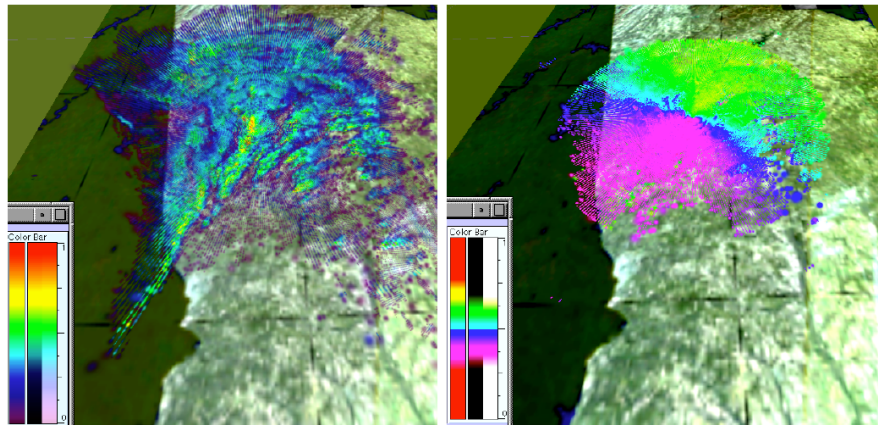


Figure 1: Doppler radar reflectivities over North Georgia (left). Yellow and red splotches (left) indicate that there is a heavy rainfall along a front from the southwest towards the center. With the property of doppler velocities, it can also be observed that the winds are heading northeast (right) [111].

This utility of knowledge products and the fact that the higher the level of knowledge, the more compact it is, are generally true. For example, the amount of published material in books in the Library of Congress is, if digitized, more than 200 terabytes [81, 101]. Presumably published books are, roughly speaking, at the highest level of knowledge since they have been digested and synthesized, written, vetted, and edited. In this sense, the Library of Congress is representative of all the knowledge in the world, at least from the point of view of Western civilization. All media in the Library (books, newspapers, film, images, sound recordings) would amount to about 3-5,000 terabytes. The book content is just a fraction of this and is dwarfed

by the amount of unpublished content we produce on paper [81, 101]. And the raw data produced are orders of magnitude larger. The Doppler radars, for example, collect approximately 1.5 terabytes of data in the U.S. per day, which would exceed the digital capacity of all books in about 130 days. And the amount of data of all forms-most of it undigested by humans-captured, stored, or replicated on digital media is estimated to be 161 exabytes (161 million terabytes) in 2006 alone. But this does not address the very much larger amount of data that was digitally observed by sensors or measuring devices of all types but wasn't captured (though perhaps a much more compact analysis, derived based on human reasoning and knowledge, was captured). So it is clear from this example that knowledge is extraordinarily compact and therefore in our visualizations, we should strive to represent relevant knowledge-based analysis products along with or often in place of raw data.

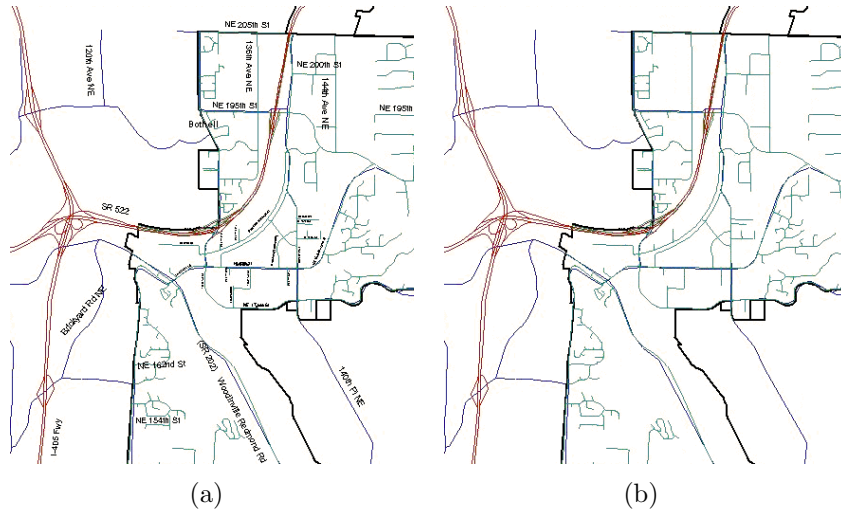


Figure 2: Street network (Kingsgate, WA) with (a) and without (b) labeling.

Although labeling is not directly connected to the definition of knowledge, labeling is important for users to build their personalized knowledge by observing displayed

labels. Figure 2 shows a street network with and without labeling. We see that an abstracted network is immediately given meaning and context by the labels. Even if one had a good mental model of an urban area and were familiar with the street layout, it would take a while and some mental exertion to understand what one was looking at and the neighborhood context. This observation extends to many other visualizations, especially (but not only) those with annotations. Of course, text labels need to be to the point and appropriate for the reasoning tasks. They also need to be legible, too many can clutter a scene, and they require attention and cognitive processing. Any design strategy needs to take these factors into account, too.

It is evident that the definition given here and the specific approach to implementing a knowledge visualization framework described in this thesis are not the only ways to define, create, and use knowledge visualizations. But the knowledge properties, design considerations, and general approach to knowledge visualizations should be broadly applicable. Whatever the approach, our perspective is that it is useful and (perhaps fundamentally) important to directly depict the knowledge structure and to link it interactively to the investigative visualizations produced for a particular problem or task. The investigator should also be able to add to the knowledge structure as she learns more about the problem. Since we emphasize exploration and knowledge discovery, there will be important and perhaps essential aspects of the task that the user does not initially know.

CHAPTER 3: KNOWLEDGE VISUALIZATION FRAMEWORK

Based on understanding the definition of knowledge, a knowledge visualization framework is designed. In *The Value of Visualization*, van Wijk [144] presents an operational model that demonstrates how the value of visualization can be quantified and calculated in a deterministic fashion. We design a knowledge visualization model with this operation model of visualization in mind, which adopts a technological viewpoint to explain how the value of visualization is measured. In his model, it is assumed that the user would continuously (interactively) create new knowledge in a visualization based on their own (or initial) knowledge. This continuous knowledge creation process is somewhat related to Polanyi's tacit knowing process of continuously understanding the underlying meaning of objects (see section 2.2 for detail). Since van Wijk's visualization model is well designed and explains the user's knowledge creation process within visualization, we begin the design our knowledge visualization model by adopting his. In section 3.1, a detailed explanation of the knowledge visualization model is provided. Since knowledge visualization mainly focuses on representing knowledge, maximizing the value of visualization is an important consideration. To determine the value of existing knowledge artifacts in a visualization, a cost/benefit analysis method is designed to maximize the benefit and minimize the cost (see section 3.2). Two distinctive types of knowledge (tacit and explicit) are adopted to design

our knowledge visualization framework based on understanding how knowledge might exist differently in visualizations.

3.1 Knowledge Visualization Model

Supplementing van Wijk’s model, we add the two types of knowledge (explicit and tacit). While van Wijk believes that “visualization is subjective” and extracting knowledge from data is an objective process that is often not considered in visualization, we propose that it is possible to incorporate general properties of knowledge (explicit) and personal properties of knowledge (tacit) in visualization. We base our modifications to van Wijk’s model on the beliefs that:

- Explicit knowledge is different from information.
- Tacit knowledge can only result from human cognitive processing (reasoning).
- Explicit knowledge exists in data, and is independent from the user or his tacit knowledge.
- Explicit and tacit knowledge are related and can be connected through the use of interactive visualization.

Figure 3 shows an operational model of knowledge visualization that includes our consideration of knowledge (explicit K_e and tacit K_t). Explicit knowledge (K_e) extracted from data (D) is represented as a visualization (V), which is received both perceptually and cognitively (P) by the user via an image (I). The cognitive processing leads to understanding and an increase of user tacit knowledge (K_t) and recursively affects subsequent perception and cognition. Tacit knowledge guides the

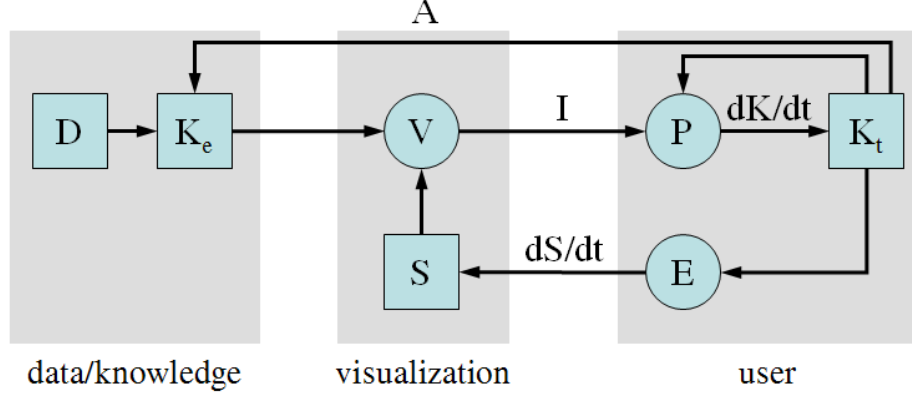


Figure 3: A model of knowledge-based visualization that integrates van Wijk's model [144] with explicit and tacit knowledge

user's interaction and exploration (E), so that the specifications (S) that control the visualization change over time.

This model shows all the main operational aspects of interactive visualization, which is to be used for the purpose of gaining insight and understanding. Further, it shows that there is knowledge internal (K_t) and external to the user (K_e) and that both should be used to gain the deepest insights. This means that the image is now a function of both data and knowledge, as well as specifications and time; this necessitates a generalization of van Wijk's equations, as indicated below. The image is perceived and understood by the user, resulting in an increase in user knowledge (K_t) and the user knowledge can be externalized as an annotation (A). The central tenet behind knowledge visualization is that the increase can be both faster and of a different order than it is with data visualization alone. (Of course, if the visualization was misleading, $\frac{dK_t}{dt}$ could be negative.)

$$I(t) = V(D, K_e, S, t), \frac{dK_t}{dt} = P(I, K_t) \quad (1)$$

van Wijk also establishes a rather insightful cost/ benefit analysis [144].

$$G = nmW(\Delta K_t) \quad (2)$$

$$F = G - C = nm(W(\Delta K_t) - C_s - kC_e) - C_i - nC_u$$

where G is the return on investment, W is the value of the acquired knowledge, and F is the profit. These quantities depend on the change in knowledge ΔK_t , the number of users n , the number of times the data are visualized m , and the number of exploratory steps per visualization k . Costs depend on the initial development cost C_i , the initial cost per user C_u (e.g., for selecting, tailoring, and learning how to use the visualization), the initial cost per session C_s (e.g., for converting data and setting initial specifications), and the perception/ exploration/ cognition cost C_e (i.e., the user must spend time watching and understanding the visualization as the specifications are changed).

This set of equations neatly sum up several main issues related to the value of visualization. For example, if development cost or initial costs per user/session (e.g., the tool is hard to use) are high the overall benefit of the visualization tool goes down. Overall, the most beneficial visualizations will be those with lots of users who use the tool often, as long as the value per session W is at a reasonable level, or those visualizations that produce a large and important increase in knowledge, even if only for a limited number of users. However, this operational approach, as far as it goes, does not tell us anything about how to design the visualization to make it most beneficial and valuable. Our approach in knowledge visualization is to look closely at

W , determine a procedure to quantify it, and establish insight into how knowledge visualizations should be designed and implemented to maximize value.

We consider the value of a piece of explicit knowledge to be a function of the relationship between the data elements within the knowledge [80]. However, this value is subject to the cost of displaying the knowledge in terms of rendering cost, clutter, hysteresis, etc. A detailed explanation how to maximize the value of visualization is included in the next section.

3.2 Maximizing the Value of Visualization

Based on the model, knowledge visualization is designed to maximize the value of visualization by determining W . As we mentioned in section 2.3, knowledge is built on relations between concepts or ideas and that knowledge is more valuable the more discriminating these relations are. For our purposes, an idea is an inference between two entities. (If no inference is made, no idea is expressed.) An ontology makes this relational structure explicit; it provides a “specification of a conceptualization [61]” in terms of concepts and logical relations.

Figure 4 shows a simple example of an ontology that illustrates its basic structure of a geopolitical ontology. The simple inferences here are all “belongs to” for different levels of the geopolitical structure. However, with this ontology, an inference engine could reason through questions such as, “what countries are within the continent of North America? or is Charlotte, NC on the state border and what state is it adjacent to?” The ontology tells us that Rock Hill is a suburb of Charlotte and also in another state (South Carolina). If the definition of the word “suburb” is clear, an inference

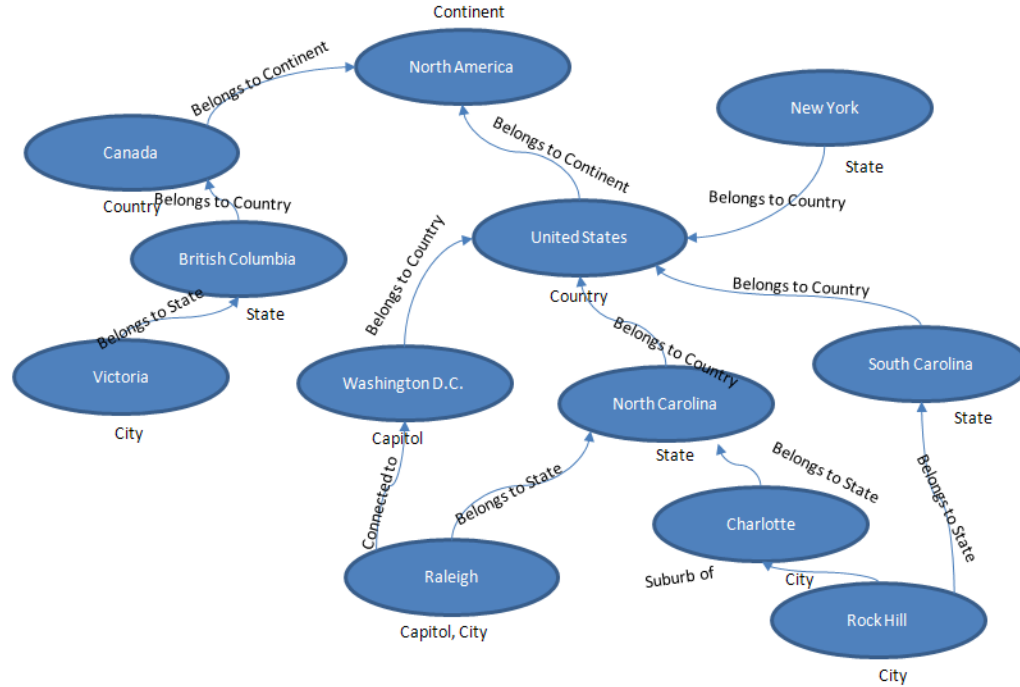


Figure 4: Geopolitical ontology for a portion of states and cities in North America.

engine could determine that Charlotte is on the North and South Carolina border. In addition, it could tell us that Raleigh is the state capitol of the North Carolina and politically connected to the capitol of United States, Washington DC.

Figure 5 represents the gene ontology (hereafter referred to as the “GO” ontology) structure of a particular gene (recombination nodule, GO:0005713). It represents the gene and gene products across all species of the gene, recombination nodule. The GO ontology is structured as a directed acyclic graph, and each term has defined relationships to one or more other terms in the same domain, and sometimes to other domains. From the GO ontology structure, it is easy for users to understand the relationship between or among gene and gene products.

In knowledge visualization, the ontology graph topology (nodes and links) can be used even if full inferential structure is not explicitly available. (It will often be

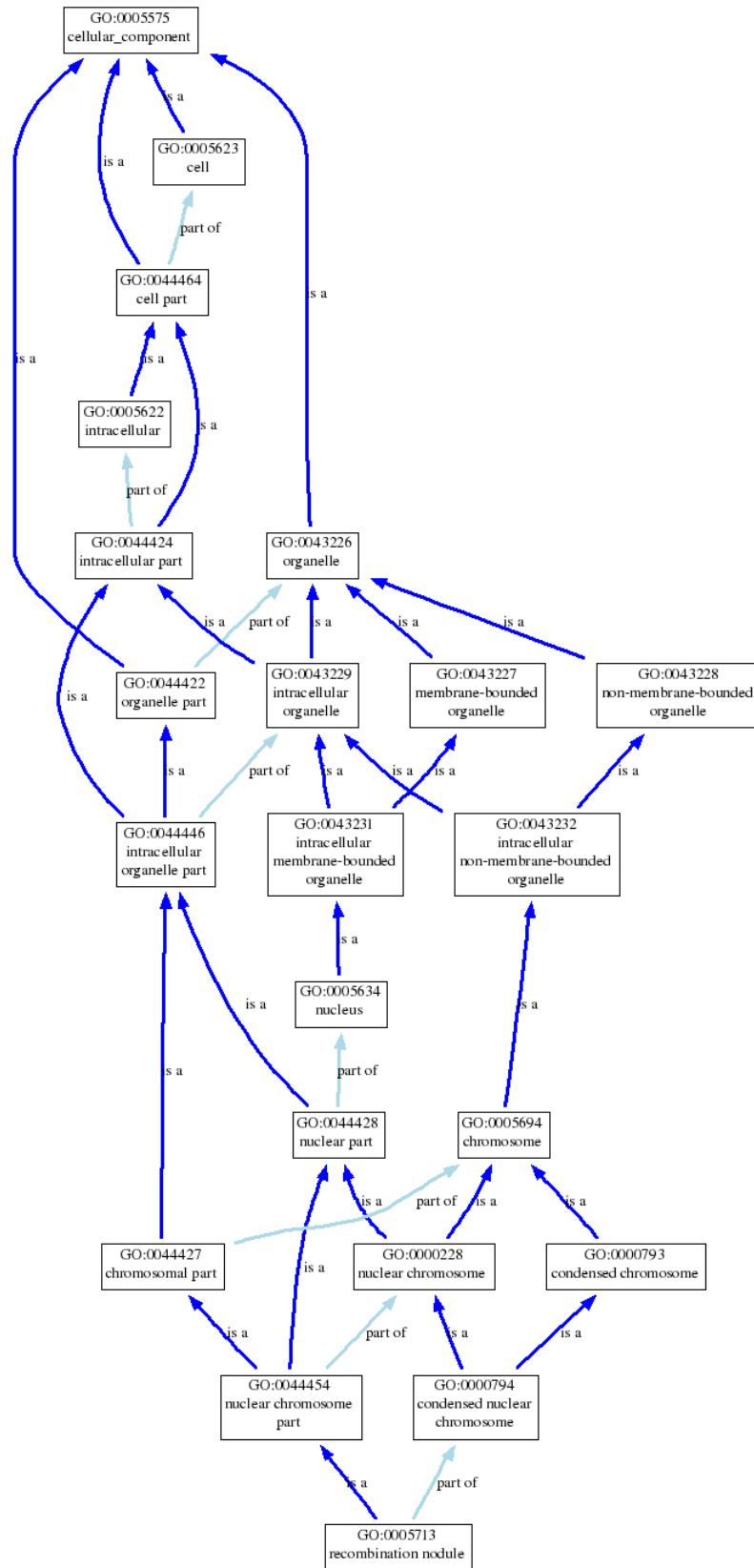


Figure 5: Gene ontology for the gene of recombination nodule [134].

implicitly available, especially if node entities are well-labeled.) In addition, we want to apply importance weights to the links; the weights could change depending on task or user interest. The reason for employing this simpler version of a knowledge structure is that we can use this approach even if we don't have a full, formal ontology. Complete ontological structures can be difficult and quite time-consuming to generate. On the other hand, there are now a variety of methods to generate relationship structures for text documents, video, multimedia, etc. For example, tools such as INSPIRE [98] generate and use keyword relations from document analyses, and Luo et al. [83] have shown how a very strong concept graph can be generated from automated broadcast news video analysis and then used in exploratory visual analyses of the themes in the news. In addition, relationship structures and taxonomies exist and have been used in many fields of knowledge. Finally, it is quite advantageous to use knowledge structures, even if incomplete, since users can add they discover new relations into the structures or can make explicit links between existing knowledge fragments. We will thus concentrate on simpler knowledge structures, although the ultimate goal of this research will be to use any structures including formal ontologies and all the inferencing that they contain.

We start with the simplest evaluation of the knowledge structure topology. That is, we assume that we have a graph of concept nodes with weighted links and then determine the nodes with the largest accumulation of weights. These are assumed to be the highest value nodes for the knowledge visualization. The calculation for node k is:

$$W_k = W_t \sum_{i=1}^R \sum_{j=1}^{N_k(i)} \sum_{l=1}^i \sum_{m=1}^{N_k(l)} Link(Node_k(i, j), Node_k(l, m)) \quad (3)$$

where R is the number of rings of neighbors included (nearest, second nearest, etc.), $N_k(i)$ is the number of neighbor nodes for ring i , and $Link$ connects the j 'th node in ring i with the m 'th node in ring l . W_t is an overall weight factor. The purpose of this equation is to sum up the contributions of all links within R rings of node k , which includes both direct links to k and secondary links to nodes that are ultimately linked to k . Conceptually, this construction is similar to the concept of centrality from graph theory, and algorithms from that field could be substituted here [13]. However, we will just use Eq. 3 here because it is straightforward and easily understandable in an interactive system. Wong et al. [152] have shown how a similar truncated analysis can be quite useful in understanding social network and other graphs and have also considered other types of topological relations. We will return to the question of other topological relations later.

Note that the weights in the above equation will change depending on the task or user focus. One way to determine user focus is by depicting the graph or knowledge structure visually and then permitting user interaction. If the user chooses certain keywords or selects by rubber banding a certain section of the graph, the weights will change accordingly. This will then result in a change in any accompanying knowledge visualization; in the former case, knowledge associated not only with the selected keyword but with associated keywords will be displayed. As discussed above, a direct depiction of the knowledge structure with interactive linking between nodes and other

knowledge visualizations and the ability for the user to add to the knowledge structure is an essential part of any knowledge visualization system. We will show this for the genomic visualization application described later. Finally, it is important to note that one does not need an explicitly linked graph to carry out an analysis. If one had, for example, a multidimensional scaling (MDS) clustering of keywords from an unstructured analysis of a large document collection [150], virtual links could then be established between keyword nodes with weights inversely proportional to the distances between them (nearest neighbors could be assigned according to weights above a given threshold), and weights could also be given to the nodes themselves based on their importance.

So far we have described a method to determine the value of individual knowledge artifacts, but we still don't have a procedure for making a complete knowledge visualization from these ranked artifacts. What we really want to do is to construct the visualization in such a way that it maximizes some perceptual and cognitive benefit. Maximizing the value of a visualization hence becomes an optimization problem of maximizing the overall value of the displayed knowledge while minimizing their cost. This relationship can be written as [79]:

$$\text{maximize } \sum_{i=1}^n b_i x_i \text{ subject to } \sum_{i=1}^n c_i x_i \leq t_c, \quad x_i \in \{0, 1\} \quad (4)$$

where n is the set of knowledge, b is the cognitive benefit, c is the cost, and x_i is the decision of whether or not a piece of knowledge K_{e_i} is to be displayed. This equation is subject to a target cost t_c , which could be the maximum resolution of a screen,

the amount of memory a computer has, or whatever the constraint is that limits all pieces of knowledge to be displayed.

Similarly, Funkhouser and Sequin [53] developed just such a procedure in terms of a cost/benefit analysis and applied it to general rendering tasks. They maximized:

$$\sum_S Benefit(O, L, R) \text{ subject to } \sum_S Cost(O, L, R) \leq TargetCost \quad (5)$$

where S is the set of objects O to be visualized, each with a particular level of detail L and rendering method R (e.g., Gouraud or Phong shading technique). Since Funkhouser and Sequin focused on rendering a complex scene based on a fixed frame rate, their benefit measure was mostly perceptual, and the cost was in terms of a target rendering time. In particular, they used the following measure (see [53] for a full description of each of these factors). Note that the *Tidy* term is not part of Funkhouser and Sequin’s equation, and will be explained later.

$$\begin{aligned} Benefit(O, L, R) = & Size(O) \times Accuracy(O, L, R) \times Value(O) \times \\ & Motion(O) \times Hysteresis \times Tidy(O) \end{aligned} \quad (6)$$

Their measure emphasizes perception in that large objects are considered more beneficial to render accurately, and if the objects in the scene are in motion, they will appear blurred to the user and thus accurate depiction is less beneficial. In the context of knowledge visualization, since we are interested in the cognitive aspects, we emphasize the knowledge-derived Value term (essentially equivalent to W), and

Hysteresis (which is a “resistance to change” factor so that a scene does not change too much from frame to frame). In addition, we have added the term *Tidy*, to control clutter.

$$Tidy(O) = 1 - Clutter(O), Clutter(O) = \frac{\min(Size_k, \sum_{i=2}^n Size(O_i) + Size(O))}{Size_k + \delta} \quad (7)$$

where $Size_k$ is the projected area of cell k that contains O , and n objects have previously been inserted in this cell. The sum starts at 2 since the first object in the cell should have clutter of zero, and the small increment δ is inserted so that the last object added does not contribute zero. *Tidy* is a quick and easy way to control clutter by using cell binning to insure focus on objects that are close to one another. It is assumed that most objects are smaller than the cell and that care has been taken so that objects are not placed on top of each other.

Choosing the optimal set of objects to render for each scene so that the benefit is maximized is an NP-complete knapsack problem [79]. However, a greedy approximation can be used to maintain real-time computation. To strongly emphasize knowledge content in knowledge visualization, we set *Size* and *Motion* in equation 6 equal to 1 for all objects.

This approach to rendering as proposed by Funkhouser and Sequin is remarkably different than the usual approach. The content of each scene is decided at rendering time, and in our case, depends not only on perceptual viewpoint but cognitive viewpoint as well. It is not even clear at the outset how well this method retains coherence for the viewer as viewpoint is changed; the amount of change over a series of frames

might be hard to control. But Funkhouser and Sequin show that the method works quite well for navigating complex 3D scenes with lots of objects (e.g., large classrooms with many chairs), and we find the same thing for complex patterns of genomic data, as described further below. In addition, in our approach, where value of displayed knowledge dominates, any object within the scene is potentially viewable, no matter what its size. There is no such thing as something being “too small to see” at a given scale. Thus in our genomic visualization application, for example, if a particular organism contains annotated gene segments that are considered quite important for the task at hand, they would be visible via glyphs decorating the organism object even in zoomed out view (where they would be much too small to see in proportion to the whole organism or in a view that shows a whole family of related genes). Finally, for knowledge visualization, it is clear that it would be very useful to have a simultaneous view of the knowledge structure itself, including a set of related concepts and keywords for possible tasks. As one interacts with this structure, importance weights would change, as described earlier in this section, and the corresponding knowledge visualization would change as well. A balanced and powerful visual analysis tool will then have coordinated windows showing the data and showing the knowledge structure for the application where interaction in one window will cause an update in the other. After showing examples of understanding the knowledge conversion processes in visualization, we are going to come back explaining how we applied the cost/ benefit analysis in Chapter 5.

CHAPTER 4: KNOWLEDGE CONVERSION PROCESSES

Knowledge Visualization focuses on representing knowledge by determining the values of knowledge through understanding the relations between concepts or ideas. Based on the proposed definitions of tacit and explicit knowledge, we provide four knowledge conversion processes in knowledge visualization: Internalization, Collaboration, Externalization, and Combination. As shown in section 2.2, these four knowledge conversions are first introduced by Nonaka and Takeuchi [97] in which they focus on how knowledge can be processed and converted from one to another in business models. However, for knowledge visualization, we propose that the four processes are also applicable because the functions of an analysts in perceptually and cognitively understanding represented information to create concrete knowledge using a visualization is similar to that of analysts in business practices. In this chapter, we present a detailed explanation about each knowledge conversion process and how each can be applied to visualization.

4.1 Internalization

In psychology, internalization is defined as the process of accepting the established set of norms, which are influential to the individual [91]. It is regarded as a cognitive process of acquiring skill and knowledge. In knowledge visualization, we propose that visually representing explicit knowledge would support analysts in understanding and

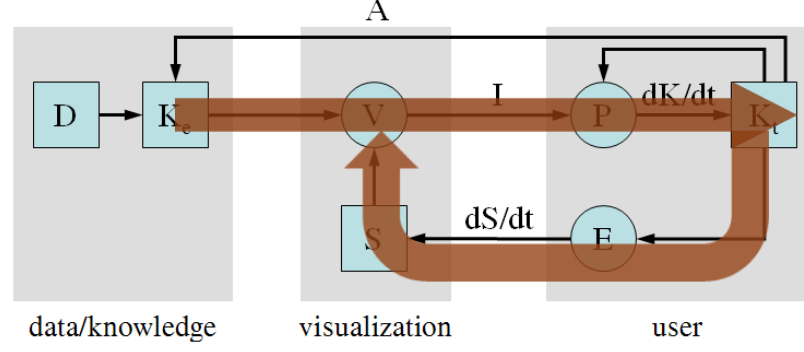


Figure 6: Internalization process (indicated by the red arrows). It explains that the user continuously builds tacit (internal) knowledge based on perceptually, cognitively, and interactively incorporating the represented explicit knowledge in a knowledge visualization.

transforming the explicit knowledge into tacit (internal) knowledge. As proposed by Nonaka and Takeuchi [97] the internalization process starts with a user discovering what the explicit knowledge is, followed by a series of steps in understanding why the explicit knowledge is of value or why it makes sense, until finally the user accepts the knowledge as their own viewpoint or internal knowledge (Figure 6). From a visualization perspective, this process parallels the concept of “insight discovery” that has been noted as the goal of visualization [26]. Since discovering insight is strongly related to building a user’s tacit knowledge based on explicit knowledge in the data, the internalization process can be thought of as the primary goal and process of using a traditional (not knowledge) visualization.

To understand the internalization process of how people build tacit knowledge from interactive visualization or visual analytics, two visualization systems are designed: Pathway Visualization (section 4.1.1) and iPCA (section 4.1.2). The two systems are designed to preserve the analytical procedures that people normally follow to analyze or solve the problems with a rich set of interactions. With such interactions,

users can interactively build tacit knowledge by looking at and interacting with the visualization.

Specifically, Pathway Visualization focuses on assisting the user in understanding gene interactions in time-series microarray data. As a preliminary step in analyzing gene interactions, the method applies two different techniques, a clustering algorithm and an Auto Regressive (AR) model. This two-pronged approach provides predictions of the dynamic pathways involved in the biological process under study. However, at this level, these two purely computational techniques lack the transparency required for analysis and understanding of the gene interactions. To overcome the limitations, a visual analysis system is designed with several visualization techniques, including pixel-based gene representation, animation, and multi-dimensional scaling (MDS). This visual analysis system allows the user to quickly and thoroughly search for and find the dynamic interactions among genes, highlight interesting gene information, show the detailed annotations of the selected genes, compare regulatory behaviors for different genes, and support gene sequence analysis for the interesting genes. In order to enhance these analysis capabilities, several methods are enabled, providing a simple graph display, a pixel-based gene visualization technique, and a relation-displaying technique among gene expressions and gene regulatory pathways. During our evaluation of this visualization, we found that people can easily understand gene interactions and find unknown, but important, knowledge.

Although the Pathway Visualization shows how people can interactively build insights, it is still difficult to understand which features can assist users in understanding the visual representation for creating tacit knowledge. Because of the reason, we

looked at a complex analytical method called Principle Component Analysis (PCA). It is a widely used mathematical technique in many fields for factor and trend analysis, dimension reduction, etc. However, it is often considered to be a “black box” operation whose results are difficult to interpret and sometimes counter-intuitive to the user. In order to assist the user in better understanding and utilizing PCA, we have developed a system (called iPCA) that visualizes the results of principal component analysis using multiple coordinated views and a rich set of user interactions (section 4.1.2). Our design philosophy is to support analysis of multivariate datasets through extensive interaction with the PCA output. To demonstrate the usefulness of our system in terms of interactively building tacit knowledge in visualization, we performed a comparative user study with a known commercial system, SAS/INSIGHT’s Interactive Data Exploration. Participants in our study solved a number of high-level analysis tasks with each interface and rated the systems on ease of learning and usefulness. Based on the participants’ accuracy, speed, and qualitative feedback, we observe that our system helps users to better understand relationships between the data and the calculated eigenspace, which allows the participants to more accurately analyze the data. We also observe that the participants build tacit knowledge easily through interactions.

With the two systems, we found that users interactively create tacit knowledge in visualization. From the Pathway Visualization, users can freely navigate through the genomic space and understand the relationship between genes interactively. Specifically, the comparative understanding between genes is especially useful for them to build the insights from the visualization. From the iPCA system, we noticed that

people have a higher accuracy when solving the given tasks than using the commercial system SAS/INSIGHT. As we will demonstrate in this section, we show that interactivity is a key factor that makes it easy for users to understand the problems and solve the tasks by building their own tacit knowledge.

4.1.1 Interactive Visual Analysis of Time-series Microarray Data

Improving medicine and human health is the primary goal of life scientists. To significantly improve medicine and human health, a more detailed understanding of the vast networks of molecules and interactions among these molecules as well as their interactions with different types of drugs is required. A more quantitative knowledge of dynamic gene regulatory pathways, i.e. the gene interactions through time, can provide an understanding of the time-dependent enhancement and suppression of gene activities and drug effectiveness. Dynamic modeling is a process through which the future of a set of variables is predicted based on the present and past values of the same variables as well as other variables. Such a modeling process not only allows the investigation of time-based interactions among physical/biological phenomena but also provides a framework to explore properties such as stability and dynamic network connectivity. Using such a model one can predict, for instance, how a particular drug can “turn on or off” a certain gene or group of genes and what combinations of drugs may be more effective over a certain period of time. However, since dynamic biological pathways involve highly complex interactions among the genes, visual representations are necessary to facilitate the exploratory analysis and understanding of these interactions.

A clustering approach is the most popular method for discovering the global relationships among gene expressions or gene interactions [47, 43]. Even though it has the advantage of having relatively low computational cost, it cannot be used to investigate and predict the time-dependent gene networks and interactions [95]. While many different methods for DNA data analysis have been proposed (see details in section 4.1.1.2), including several for microarray data, there is little work that has been done on detailed analysis of time-series DNA microarray data [156].

In this section, we present a new visualization-based framework with emphasis on temporal feature display and visual analysis on time-series microarray data. In particular, the framework applies a gene regulatory pathway (e.g. gene regulatory network) prediction method to predict the gene expressions over time. Although similar in nature to the combined models in [92, 82], our method applies a clustering method to group the gene pool into a number of biologically-meaningful clusters. Then an Auto-Regressive (AR) model is used to relate the expression levels of each prototype in time t to the expression levels of other prototypes in previous times. The AR model is one of the broadly used linear prediction methods that predict the output of a system based on its previous values [39, 40].

Based on our combined model (a mixture of clustering and AR model), the visual analysis framework has been designed to support finding known and unknown gene interactions as well as understanding individual genes in detail (The overall layout of the visual analysis system is shown in Figure 7). In the framework design, we define two steps: *Visual Representation* and *Visual Analysis*. Although both are broadly used terms in the visualization domain, we define them as follows: *Visual*

Representation serves the major role of displaying visual entities in a visual space (2D or 3D) and *Visual Analysis* supports the user’s interactive understanding of visualized entities and their relationships (see section 4.1.1.3 for detail).

In the following subsections, several design properties and techniques used in the framework will be described further:

- Biological pathways and prediction techniques used in our model,
- Representation methods on gene interactions and clustered pathway information over time,
- A novel overview and detail-view approach with which the user can employ interactive analysis to create new understandings of dynamic pathways, gene properties, and relationships,
- Design schemes used in the detail view to continuously reveal the patterns and relations using simple graph drawing and pixel-based representation techniques,
- An integrated analysis tool that can be launched and whose results can then be compared.

That this framework is both efficient and effective in the analysis of time-series microarray data as a complex multi-step process, is supported by the insights described in section 4.1.1.8.

4.1.1.1 Related Work

There are a few studies focusing on the temporal regulatory properties of microarray data. Some such approaches include: correlation analysis [47], Boolean Networks

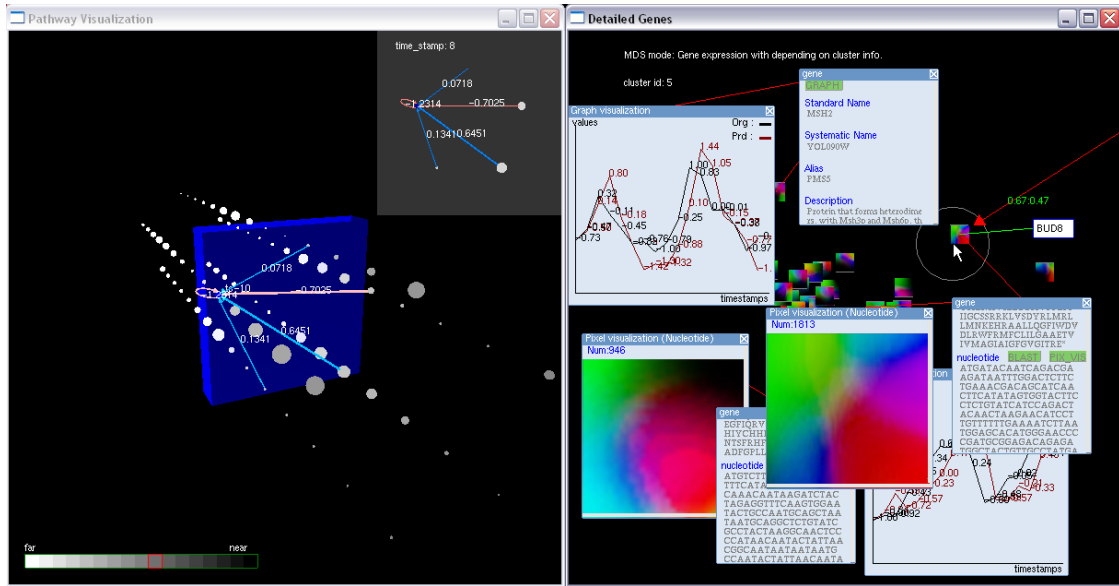


Figure 7: The overall layout of the visual analysis system, which consists of two main windows: one (left) is for showing regulatory pathway information depending on time, and the other (right) is for providing interactive analysis of gene information using pixel-based and line-graph visualizations and textual annotations.

[125], Bayesian Networks [52], Dynamic Bayesian Networks [100], etc. These methods have proved to be useful in finding the regulatory network for certain applications. However, considering the small number of time steps in typical microarray time series and the large number of genes involved in many biological processes, the parameters calculated in these methods may not be reliable [156]. Instead of using computationally complicated approaches, several researchers addressed the efficiency of using signal processing models in understanding time-series microarray data [125, 39].

Although statistical methods and signal processing models are useful to understanding gene interactions and regulatory events, it is still difficult to utilize all features in microarray analysis. To address this limitation, several pathway visualization applications supporting the understanding of the functions of genes have been designed. The most broadly used applications are KnowledgeEditor [139], GenMAPP

[38], GeneSpring [124], PathwayStudio (Formerly known as PathwayAssist) [9], etc. KnowledgeEditor is useful to model and analyze biological pathways directly creating biomolecular network graphs in order to find molecular interactions, but it has no prediction capability. GenMAPP is freely downloadable and widely distributed. It visualizes gene expression data on maps representing biological pathways and groupings of genes. Commercial applications such as GeneSpring and PathwayStudio are efficient at permitting the user to create her own gene pathways and find interconnections between genes. Genespring in particular is designed to produce scatter plots as well as correlation values. Although these tools are useful for certain microarray analysis, they do not fully support understanding of gene interconnections in highly interconnected pathways or analyzing time-series microarray data. Because of this limitation, several researchers tried to design visualization applications supporting the analysis of time-series microarray data.

Craig et al. [35] introduced a technique which displays all continuous temporal features of microarray data. They designed a tool (called time-series explorer [36]), with which users easily can find similar and important patterns on time-series data. Peterson [102] addressed the difficulty of analyzing time-series microarray data by proposing a technique which changes the coordinate system using principal component analysis (PCA) in order to help in observing dynamic behaviors of the data. Symeonidis et al. [129] designed a technique visualizing clusters of genes as a crossing-free graph. Even though their work is good in understanding and finding the important patterns on time-series microarray data, they have not addressed the needs of predicting the pathway values and analyzing gene sequences.

Due to the lack of suitable logical connections between existing analytical tools and the visualization tools, the use of these visualization tools has not proved as insightful as it could be. However, evaluation has been applied to provide the basic requirements in considering HCI (Human Computer Interaction) methods for pathway visualization systems [116]. Unfortunately, many of such requirements have not been followed in existing visualization applications. Eisentein [48] described several technical features that should be in microarray analysis applications and pointed out that “the major objective of microarray experiments is not to generate endless spreadsheets and scatter-plots, but to produce data that can be used to formulate an understanding of biological events.” In designing our visualization system we have kept in mind all these requirements and guidelines in order to maximize the user’s understanding of biological interactions.

We use a signal processing model by combining an Auto Regressive (AR) model [44] and a clustering method. In contrast to other methods, our model is useful in analyzing and predicting the dynamic behaviors of time-series microarray data. Even though the method is designed to predict the signal patterns and dynamic interactions, understanding and analyzing the information produced by the model may not be straightforward. In particular, concepts such as cluster prototypes, time steps, and excitatory/inhibitory interactions need to be effectively presented to users [156]. To address these needs, we use interactive visualization techniques to find important patterns, preserve temporal features of time-series microarray data, and analyze gene expressions interactively.

4.1.1.2 A Dynamic Model

To develop a dynamic model of gene regulatory interactions, we have created a combined technique consisting of Auto Regressive (AR) modeling [44] and a clustering method. Since almost all biological pathways are composed of a large number of genes, the approach of applying the AR model directly to the individual genes is not computationally feasible. That is, the number of genes in a biological process is often so large that it is impossible to develop reliable AR models for the typically short time-series microarray data while directly incorporating all genes as individual AR variables. To address this issue, we exploit the fact that many genes behave very similarly in the biological sense and therefore can be clustered together before dynamic modelling.

As a pre-processing step, K-means clustering is used to group the gene pool into a number of biologically meaningful clusters. Each of these gene clusters is represented by a prototype that reflects the overall time trends of the cluster. After the preprocessing step, the AR model is applied on each cluster as opposed to individual genes. Since the number of clusters is small enough, the AR model can be reliably developed for the prototypes. The model relates the future expression level of the prototype clusters to the values of the prototypes in past time step (t). The model also considers uncertainty inherent to the model by considering a noise factor (e). In its most general form, the model is a linear system of difference equations:

$$\begin{aligned}
y_i(t) = & -a_{i11}y_1(t-1) - \dots - a_{i1n_{1i}}y_1(t-n_{1i}) \\
& -a_{i21}y_2(t-1) - \dots - a_{i2n_{2i}}y_2(t-n_{2i}) \\
& \dots \\
& -a_{ip1}y_p(t-1) - \dots - a_{ipn_{pi}}y_p(t-n_{pi}) + e_i(t)
\end{aligned} \tag{8}$$

where $y_i(t)$ is the expression level of prototype i at time t , n_{ji} is the maximum time span of the interactions between prototype of cluster i and prototype of cluster j , coefficients a_{ikj} 's are the parameters of the model, and $e_i(t)$ is the noise factor.

In here, we use a dataset containing the time-series expression values of approximately 200 genes involved in the cell cycle of the budding yeast *S. Cerevisiae* [29]. The gene expression values were collected in 17 time points. It is known that there are five major phases in cell cycle development: Early G1 phase, late G1 phase, S phase, G2 phase and M phase. In each phase only genes whose biological functions correspond to the changes occurring in that phase are active. Based on the known biologically-distinctive functions of these five phases, it is reasonable to cluster the genes into five clusters. The results show that the model can very accurately predict the expression values of almost all genes in the future steps (see [39] for detail).

The main application of this dynamic modeling method is twofold. First a dynamic regulatory network governing the quantitative interactions among the prototypes of the main biological trends can be obtained, i.e. the model discovers the effects of each gene group on itself and on other groups in time. Secondly, by using the resulting

dynamic network, the expression level of each gene at time t can be predicted based on its expression level and expression level of other genes.

4.1.1.3 Designing a Visualization Model

We define several major steps to be followed and use them to form our system (Figure 8).

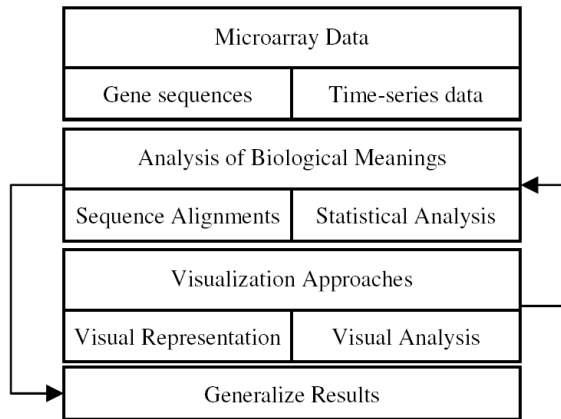


Figure 8: Major steps in visual analysis of time-series microarray data.

In microarray data analysis, one must find the biological meanings of the discovered genes. To support finding the biological meanings, two different approaches, gene sequence analysis and statistical analysis, are typically used. Sequence analysis is performed on the gene sequences to find similar sequential patterns or regulatory sequences, some of which have been previously studied and annotated. Statistical analysis is done using microarray data to identify the regulatory networks or cyclic genes. While statistical analysis can be performed alone without understanding the gene sequences, it is still necessary to consider the gene sequences because sequence analysis can assist in finding the most related genes (regulatory sequences) to the biological system and provide the molecular basis for understanding the detailed gene

interactions. Therefore, gene sequence analysis and statistical analysis on microarray data often have to be performed together to understand gene function in detail [127]. This process is usually followed iteratively until relevant knowledge is found. The knowledge must then be generalized to help other research scientists understand fully the meaning of the genes; identified cyclic genes, regulatory events of activation or inhibition, etc.

In particular, we design methods supporting each analysis in Figure 8 and the overall iterative process. Specifically, we define two steps as *Visual Representation* and *Visual Analysis* in our system.

Visual Representation is a preliminary step which can provide the user a general understanding of biological pathways in display space. Since biological pathways are highly complicated, we use both *2D* and *3D representations*. These representations are designed for interactive analysis. In the 3D view, the analyzed data (clustered pathway information) are directly mapped to the visual attributes, while interactive analysis of the genes is shown in the 2D view. Temporal attributes in time-series microarray data are carefully managed by preserving clustered gene expressions in the 3D view as well as each gene's individual expressions in the 2D view. The major advantage of using a 3D representation is to increase the possibility of integrating additional visual relations and information into the representation [138]. *Animation* is added to maximize efficiency and create a better understanding of data [112] because it creates an additional display dimension [154]. However, 2D representation has an advantage of understanding a gene's detailed biological meanings. *Pixel-based gene representation* and *Positioning* are applied to support discriminating a gene's

properties and understanding its relationships respectively.

Visual Analysis is necessary to support exploration of interactions among clusters, to increase the capabilities of both exploratory and focused analyses of gene interactions, and to overcome the limitation of cluttered displays. Four different *visual interactions* are used to facilitate analysis: *labeling*, *detailing*, *navigation*, and *focusing & linking*. *Labeling* displays a simple description in areas of focus. It helps in discriminating the information in highly cluttered display. *Detailing* interactively provides the user with enriched knowledge when requested. *Navigation* gives the user the ability to move through the visual layouts at different scales or extents. *Focusing & linking* allows comparison between genomic data to find similar patterns [18].

Based on the above considerations, the visual interface is designed as shown in Figure 7. It has two windows. The user starts with exploration of the cluster time series in the 3D view. As the user focuses on time steps, clusters, or individual genes, the 2D view is dynamically updated. The user moves back and forth between these windows as she explores and analyzes. Detailed descriptions of visualization approaches as well as this highly interactive, iterative process are provided in following sections.

4.1.1.4 Visual Representation

4.1.1.4.1 Visualization of Clustered Gene Expressions

Most microarray visualization applications use a visual form (e.g., glyphs) to represent abstract information. But there are limitations to what can be accomplished due to limited display space and the scale and richness of detail of the data [116].

Several information visualization techniques are relevant to this problem, such as Overview+detail [138], Focus+context [106], and Pad++ [11]. Multi-scale visualization techniques are appropriate to manage large and complex microarray data [116]. Following this path, our framework uses multi-scale visualization [65] to effectively and efficiently manage and increase understanding of the large-amount of microarray data. In this section, we will explain in detail how we use this framework for displaying microarray data.

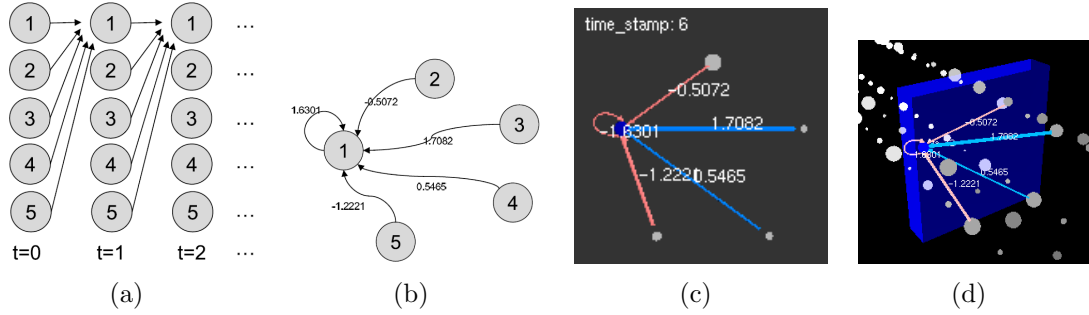


Figure 9: (a) Sketch of time-series gene networks for analyzing cluster number 1; (b) static graph prototype at 6th temporal point; (c) direct mapping onto 2D; (d) 3D mapping in context of all time steps.

A 3D representation method is used to show the time-series microarray data. However, since the 3D display can produce clutter, the selected gene expression values are laid out simultaneously in 2D (Figure 9). The 3D representation is efficient in showing the gene expressions at the successive time step where the current expression level of prototype i at time t has a regulatory relation (activation or inhibition) with the prototype j at time $t - 1$. Each prototype is regarded as a cluster and represented as a 2D circle or 3D sphere respectively in 2D or 3D view. The perspective view in 3D representation permits the user to maintain an overall context in time while exploring dynamic interactions in detail.

The radius of the cluster represents the normalized mean value of the clustered signal pattern. Thus a bigger sphere or circle represents stronger overall contributions of the cluster to the regulatory network at that time step. Also each cluster makes use of a color coding [119] to enhance visualization of cluster values. When selected, a cluster changes color to blue. Transitional arrows colored blue indicate positive transitional pathways (*activation*) between cluster spheres, and arrows colored red indicate negative transitions (*inhibition*). That is, the transitional arrows show controls of one cluster over another with blue indicating enhancement of gene expression and red indicating suppression. The self-transitional loop indicates a control of the cluster on itself. With this simple palette, users can quickly discern important pathway differences and the state of the regulatory network over time. The user can also select individual arrows for quantitative information.

As shown in Figure 7 (left), the layouts are accompanied by a temporal slider bar at the bottom left that supports animation corresponding to time-series steps, in particular dynamic animation of cluster pathways over time. The slider bar ranges over all time steps in the dataset. In the bar, the time series is colored along a gray scale from white (17th time step) to dark gray (1st time step). As Wright [154] points out, animation can be an additional dimension to support displaying and analyzing information datasets in 3D view. Also, it provides a natural mapping between time and saturation that allows the user to identify positions in the temporal space while permitting color to be used for other information display. When the user makes a selection on the bar, a thin blue semi-transparent box appears at the selected time step. If the user drags the mouse along the slider bar, the box moves accordingly

along the time steps. Alternatively, the user can grab and drag the box directly. In either case an animation showing changes over time is produced. In addition, selections can be made within the box (e.g., a cluster is selected and its transitional arrows displayed). These selections are then maintained at later or earlier time steps as the box is moved around. By employing this animation capability, the user can quickly analyze and compare behavior in time with minimal effort [112, 17].

4.1.1.4.2 Visualization of Individual Gene Expressions within Clusters

As mentioned earlier, finding biological meanings is important in microarray data analysis. Even if the visual layout described above gives the user a quick overview of the dynamics of the gene regulatory network, significantly more details (gene sequences, interactions, regulatory behaviors, etc) are needed to understand each gene. In general, gene (or protein) sequences range about a few thousand units in length. To manage data for a single protein sequence and provide contextual pattern information in a useful way, a pixel-based gene representation method is used. To support a better understanding on gene interactions and regulatory behaviors, MDS [5] is a useful technique to effectively represent high dimensional data in lower dimensional space. In display space, each gene is positioned depending on its interactions and regulatory behaviors. All these representations are laid out in their own sub-windows in the right window of Figure 1. The user can select and bring forward the particular sub-windows desired for current exploration. Additionally, several interactive analyses are provided to increase the understanding on time-series microarray data and gene interactions. In this section, techniques used for designing this framework are

described in detail.

4.1.1.5 Pixel-based Gene Representation

The pixel-based gene representation is designed to reveal the features of gene sequences. To map from gene sequences to pixels, space-filling methods [77] are quite useful because they produce spatial patterns that have consistent locality, even for long gene sequences. Several methods of this type have been designed. Here we use a Hilbert curve ordering method [16] to arrange sequence information mapping with color information, since it has the advantage of providing continuous curves while maintaining good locality of sequence information.

For mapping with gene sequences, we set the Hilbert curve order to 12 which covers $2^{12} \times 2^{12}$ sizes of gene sequences. Color coding is then used to represent the sequence information. DNA is a linear polymer made up of sequences of four nucleotide bases: adenine, guanine, cytosine, and thymine - designated A, G, C, and T. Gene regulatory pathways can involve hundreds or more genes from which different proteins can be expressed. Hence, two different color mapping approaches have been made, one for the gene sequence and the other for the expressed protein. Originally, pixel-based visualizations on large scale of data were pioneered by Keim et al. [78, 77]. And a pixel-based gene representation method has been suggested by Wong et al. [153]. But they did not concern themselves with finding an efficient color coding for the gene sequences.

For determining the correct color codes, four commonly used color maps used by other researchers were tested on the gene transcription complex *SWItching deficient*

$(SWI4)^2$, a protein involved in the budding yeast *S. Cerevisiae*, in order to find the best-mapped color codes. All images in Figure 10 are generated using Hilbert curve ordering.

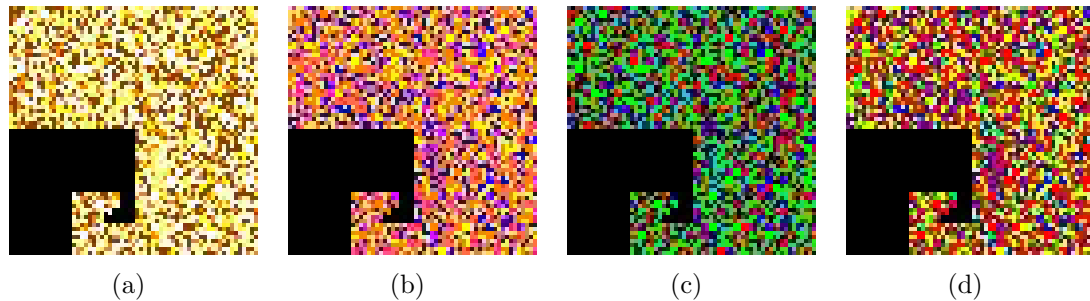


Figure 10: Pixel-based gene (*SWItching deficient (SWI4)*) representation with several different color codings; (a) A (white), C (yellow), G (orange), T (dark brown) [153], (b) A (orange), C (blue), G (purple), T (yellow) [114], (c) A (green), C (blue), G (black), T (red) [113], and (d) A (red), C (blue), G (green), T (yellow) [110]. Black space located in left bottom of each image represents the empty part of the space-filling curve.

Figure 10 shows mapped images with different color codes. But it is difficult to understand clearly the represented patterns and features in the sequence. Therefore, a pixel enhancement technique (initially proposed by Wong et al. [153]) is applied in order to find the best color mapping method. The pixel enhancement technique consists of three steps. First, a Gaussian filter is used to smooth the high-frequency values. And then histogram equalization is applied to modify the dynamic range and the contrast of an image depending on color channels (for example, R, G, and B channels). Finally, saturation values are increased using extrapolation as a saturation adjustment technique.

Even though all pixel-enhanced images have similar results (Figure 11) with respect

²SWI4 acts as a transcriptional activator to regulate late G1-specific transcripts in budding yeast required for DNA synthesis and repair.

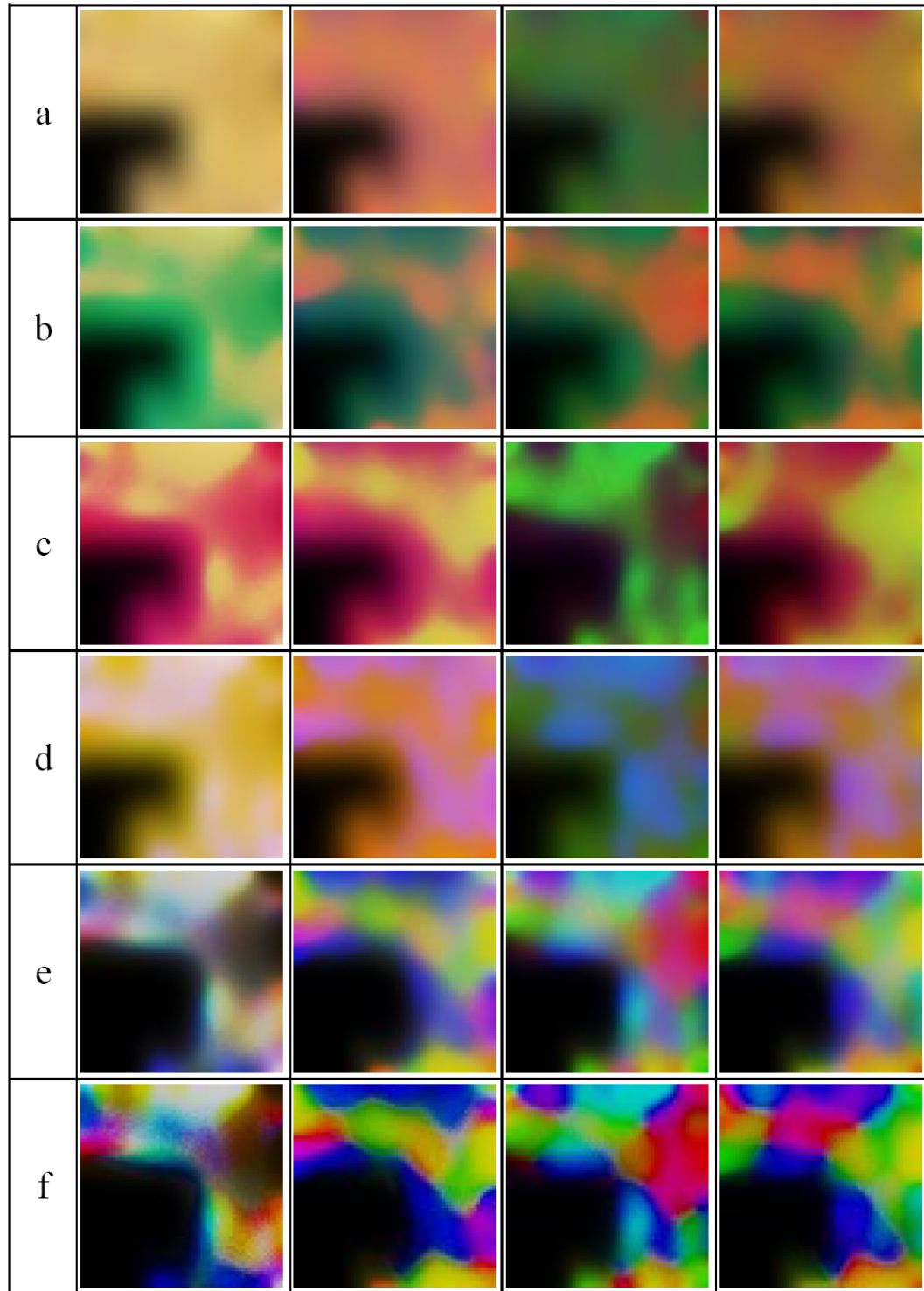


Figure 11: Rows represent several image processing filters and a pixel enhancement method are applied to pixel maps shown in Figure 10 (represented by columns). (a) Gaussian filtering, (b) Histogram equalization on red channel, (c) Histogram equalization on green channel, (d) Histogram equalization on blue channel, (e) Merged all color channels, (f) Saturation adjustment is applied with the saturation value of 2.5

to the patterns in terms of R, G, B color channels, the pixel representations in the 3rd column of Figure 11, using color codes from (c) in Figure 10 (A (green), C (blue), G (black), T (red)), show consistent patterns in all 3 channels. Furthermore, merging all channels and applying saturation adjustment (Figures 11(e) and 11(f)) show in clearest detail which area of the image has denser information for adenine, guanine, cytosine, or thymine, and what its shape is. We thus apply the result of Figure 11(f) in the following. To arrange the gene images in 2D view, we must next apply MDS, as discussed next.

It is important to note that this is a simple way to construct gene images. More sophisticated methods, which might take into account sequence structure and meaning, could be applied. However, the pixel mapping would be the same.

4.1.1.6 Data Positioning in 2D View

In our framework, we use nonmetric multidimensional scaling [143]. Based on this method, the dimensions of time-series data (17 dimensions of timestamps) and gene sequences (4 dimensions of nucleotide bases) are changed to two dimensions.

In each case, the first step is to find a monotonic relationship between the dissimilarities. The relationship is typically found using isotonic regression. In our application, we use Kruskal's nonmetric phase [143] as isotonic regression. For the case of gene sequences, we opted for a simple approach that references the numbers of nucleotides when finding the relationship. First, gene sequences are counted in terms of adenine, guanine, cytosine, and thymine. Based on the counts, the relationship is determined. Finally, all gene sequences are mapped onto 4 dimensional distance

matrixes and MDS is applied to map pixel-represented genes (glyphs) to a 2D display space. Even though more detailed measures of relationship on gene sequences could be applied, this approach is quick and gives a rough idea of contextual similarity.

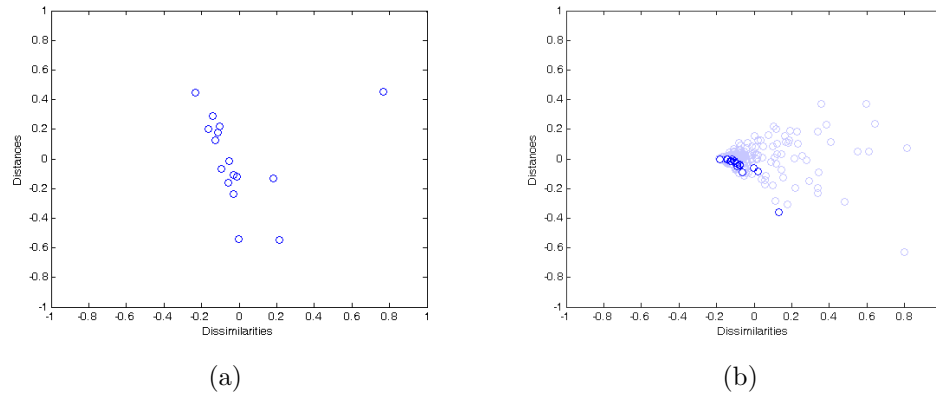


Figure 12: Local similarities of time-series data within a cluster group (a) and global similarities of all time-series data (b). Each circle indicates time-series data of each gene and the highlighted circles in figure (b) indicate the data having the same clustered group in figure (a).

Two different MDS representations are considered with respect to the time-series data. One representation shows local differences among genes in the same cluster group (Figure 12(a)), while the other shows global differences among all genes in all the clusters (Figure 12(b)). Selections in Figure 12(a) are then highlighted in Figure 12(b). In this way the user always has a view available that emphasizes local and global changes over time. The user can then switch attention to a more detailed view for a selected time step (Figure 13), where the circles are replaced by glyphs.

In Figure 13, two different glyph forms are used, the pixel-based representation of gene sequences and a simple line graph representing the time-series microarray information. As described above, our visual analysis method is closely tied to the dynamic pathway prediction model. The line graph display for each gene, where

the gene expression level is plotted along the vertical axis and the time steps along the horizontal axis, provides an option that emphasizes each gene's time history. Distances between genes are then measured according to the similarity of their line graphs. Hence, there are 4 possible ways to construct gene expression patterns, as indicated in Figure 13.

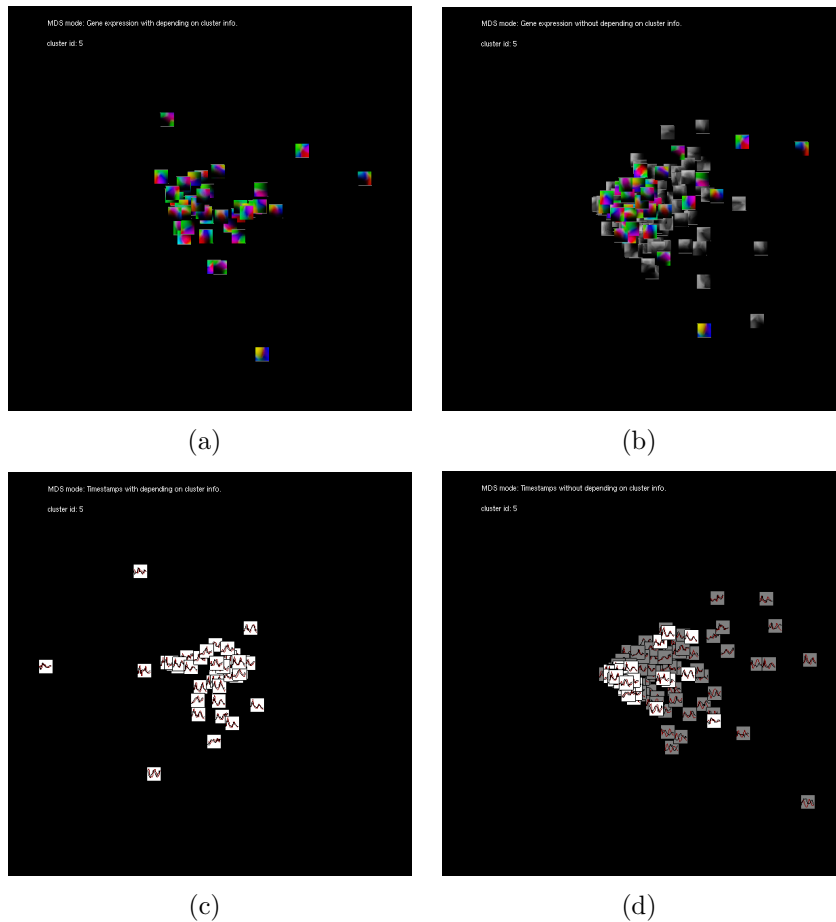


Figure 13: Glyph representations of (a, b) pixel-based gene sequences and (c, d) graph-based DNA microarray time-series information. Both local analysis (a, c) and global analysis (b, d) are given. Highlighted glyphs in (b, d) indicate the genes located in the selected cluster with respect to other genes.

From Figures 13(a) and 13(b), it is seen that spatial distributions of the local and global gene representations are not much different. But the spatial distribution of the graph-based representations (Figures 13(c) and 13(d)), show major differences

between local and global analysis, with the latter case producing highly clumped results. It is thus clear that clustered data should be analyzed for both local and global patterns, considering both gene sequence (pixel-based glyphs) and time series (line graph representations), since these bring out different aspects that can be used to bring out differences and similarities in gene structure, function, and regulatory behavior. This point is illuminated further in section 4.1.1.7. Visual Analysis methods discussed next enrich these capabilities.

4.1.1.7 Visual Analysis

Interaction is a very useful exploration and discovery tool for large scale or complex data. In our visual analysis framework, we apply several interaction methods such as *labeling*, *navigation* (panning, zooming, and scaling), *detailing*, and *focusing & linking*. In this section, the interaction methods are described.

Labeling To heighten the knowledge content of the glyph representations in Figure 13, we need to have a way of showing brief annotations for specific genes such as scientific name, gene expression values, regulatory events, etc. The method of excentric labeling has been explored for textual labeling [50]. It uses the technique of directly attaching a focus region to the cursor, and annotations are shown whenever the cursor is passed over glyphs. Although several different excentric labeling techniques have been proposed [50], there are still some drawbacks with respect to *cluttering* due to annotations. Also, the method is not able to show the gene regulatory events, and context may be lost since the annotations appear and disappear as the cursor is moved. Therefore, we have designed a modified labeling technique which adopts

the position-shifting operation [30] to show textual information without *cluttering*. Manual position shifting is a broadly used technique because it is especially useful in controlling labels in dense information visualizations. But it requires user efforts to manually shift the annotations. Instead of using the manual operation, our approach uses an automatic shifting method. The size of the overlapped region between labels is measured, then, using a strength based on this size, each label has a repulsive force against any overlapping label, so that the labels are pushed apart. This reduces the need for user attention and effort and permits the high value annotation knowledge to flow unimpeded to the user.

The labels can contain a variety of information. Here the labels contain gene names and, in the case of the line graph representation, arrows indicating the point in the line graph corresponding to the current time step (Figure 14) which supports analyzing the microarray time-series details and relations.

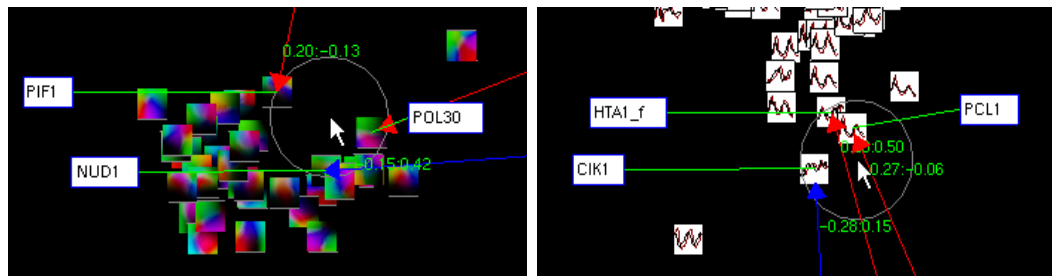


Figure 14: A modified excentric labeling provides labels in a focus region. The focus region dynamically changes its location while the cursor moves over the display. Labels of the objects located in the focus region are updated smoothly and dynamically depending on the focus region.

Additionally, the labeling technique supports changing the size of the focus region centered on the cursor, with which it is useful for showing labels when the scale of the glyphs is modified or when the user is interested in a larger or smaller region. The

effective rearrangements and smooth transitions caused by the force fields help the user maintain context and keep track of new annotations even under continuous cursor movement. All these capabilities are incorporated in the navigable view (Figure 15).

Navigation Zoom and pan navigation within the detail view is designed based on the “Pad” [99] metaphor and its extension, Pad++ [11, 54]. In this metaphor, the visual space is considered as an infinite 2D plane (called Pad), which can be stretched by orders of magnitude at any point to investigate details. In our previous design of a genomic visualization system, GVis [65], we found that the technique provides an important capability for finding details and relations at all scales within a context of thousands of genomic data.

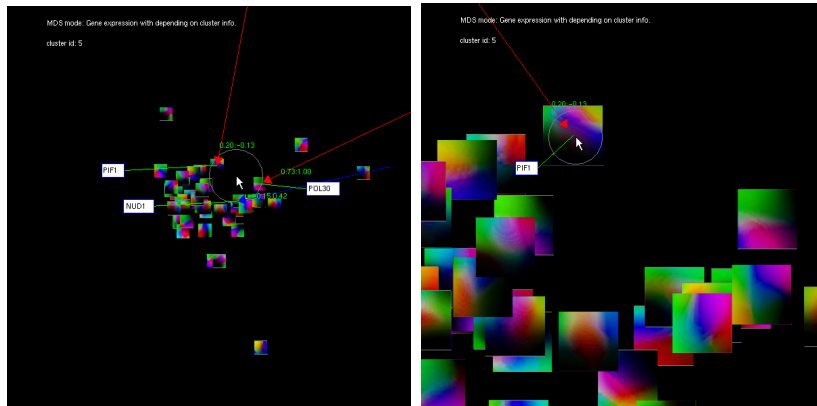


Figure 15: Navigation in the zoomable space provides the capability for smoothly finding and examining interesting data objects in overview (left) or close-up (right).

By using the Pad++ metaphor, a user can easily compare objects’ patterns and find their differences over the whole object space and at multiple scales through successive pans and zooms. Figure 15 shows zoomed-out and zoomed-in navigation states. When the glyphs are in the zoomed-out state, we can find the overall arrangement of the gene expressions with respect to each other. Upon zooming in, we can take a close

look at the various glyphs to find complete details in the gene expressions.

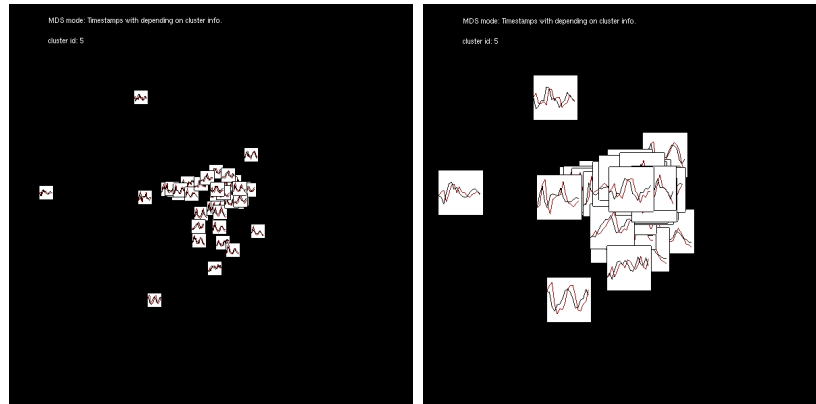


Figure 16: Scaling makes it possible to see the detail of glyphs without disrupting their location information.

Scaling is included in the navigation interactions and is quite useful since it shows the glyphs at larger or smaller scale without changing their positions or the viewpoint and retaining the overall pattern. Scaling can avoid or reduce overlaps, or increase glyph sizes to show details, as shown in Figure 10. Even though navigation and scaling work well in avoiding overlaps in the display space, there are still cluttering problems where elements are highly correlated and thus close to one another. To minimize this problem, we permit the reordering of overlapped glyphs in display space. Selected glyphs can be moved to the front or back with respect to the user's viewing perspective. In this way a clear view of the details of a glyph can always be obtained.

Detailing Since genomic data may contain large amounts of knowledge on sequence, active sites, expressed proteins, etc., it is necessary to present this information as needed. For best focus and efficiency, we provide these annotations in detailing windows within the display and attached to the relevant gene glyphs. These pop up in a

“details on demand” mode, upon selection. (See Figure 17(a)).

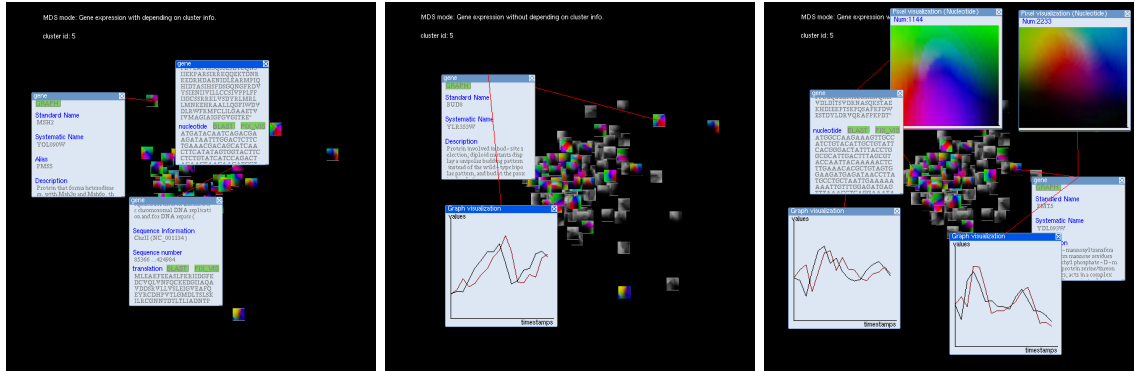


Figure 17: Representations of information about the selected genes. (a) Additional information will be given for further understanding of the selected genes. (b and c) pixel-based gene representations and line-graph visualization are used at the same time when analyzing genes.

Each window has a standard format for the text record of the item, including the standard name, the original DNA reference name, the sequence start and end, the gene’s function, the proteins it creates, and so on (Figure 17(a)). By panning or scrolling within the pop-up window, text of any length is accessible to the user. The windows align themselves dynamically in the 2D space, near the glyphs they represent, so that overlap is minimized as several boxes are opened. Each window can be positioned into another location by dragging it. It also can be scaled up or down to show the text information more clearly or to avoid overlaps.

Focusing & Linking To support gene analysis, the pop-up window has buttons for pixel-based visualization, line-graph visualization, and BLAST. The BLAST button launches the sequence analyzing tool called BLAST (the Basic Local Alignment Search Tool), as described further below.

Figure 17(b,c) shows that this provides a fast and comprehensive view of the avail-

able gene sequence information permitting, for example, comparison between pixel-based and line graph views for any gene or gene collection. This close comparison can extend to two or more gene sequences.

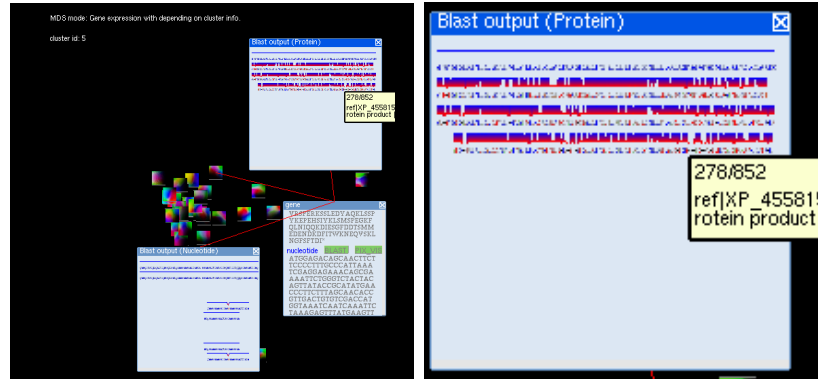


Figure 18: The BLAST outputs are shown with protein and nucleotide sequences (left); a close-up of BLAST output (right) shows similar sequences aligned for comparison. The window has a capability of scaling the output to show in detail.

As mentioned above, our application incorporates a widely used sequence matching tool, BLAST, which detects relationships among sequences that share only isolated regions of similarity [8]. The National Center for Biotechnology Information (NCBI) provides BLAST utilities such as network-based BLAST, stand-alone BLAST, etc. When the BLAST button is clicked, network-based BLAST is launched and searches for similar alignments in the NCBI database. Depending on the option selected, results are provided either in terms of alignments with the selected nucleotide sequence or its expressed proteins, which are then displayed in the pop-up window (Figure 18). The outputs are aligned in a graphical representation that permits the user to get overviews or detailed comparisons by zooming in or out. This is especially useful when finding regulatory elements among genes that were indicated in the time series line graphs.

To complete the interactive analysis framework, we have incorporated the GVis system [65]. GVis supports in-depth study of the structure and function of the organisms, genes, and their expressed proteins involved in the dynamic regulatory pathways. This study provides the user with a natural, more detailed follow-on to the fast exploratory analysis carried out with the above interactive tools. Thus GVis automatically takes genes selected from the above analysis and provides detailed comparison of their fully annotated gene environments with any other genome environment (which may be selected by iterative applications of BLAST or other comparative analysis tools). GVis is capable of permitting interactive exploration of tens of thousands (or more) of genomic data from overviews down to the level of the annotated nucleotide sequences.

4.1.1.8 Building up the Biological Insights

In this section, we explore what kinds of biological insights can be gained when using our framework by presenting some results from visual exploration of the microarray analysis of gene regulation in budding yeast *S. Cerevisiae*.

The user starts by exploring the time series data using the left side of the interface in Figure 7. When she focuses on particular genes, the predicted regulatory pathways are displayed on the right side of the interface in Figure 7 in order to provide a detailed view of dynamic gene regulation. As the user zooms in, the pathway values are automatically shown when the scale of the window is large enough to display them on screen. Figure 19 shows successfully predicted regulatory pathways alongside the actual observations. This demonstrates that the predicted pathways can be used even

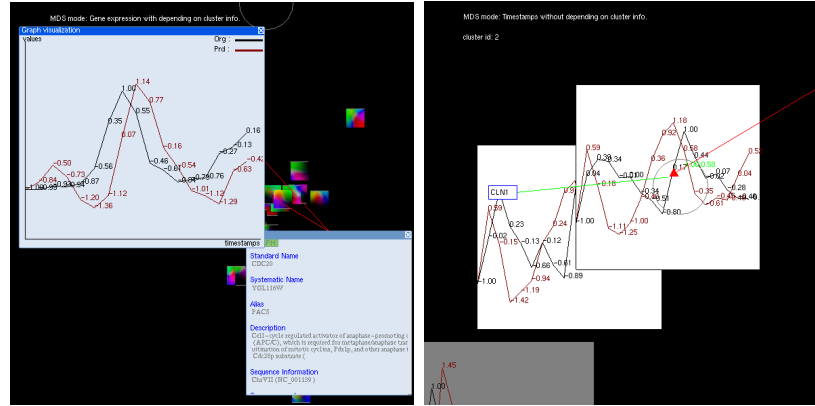


Figure 19: Two different ways of representing methods: gene expressions of CDC20 (*Cell Division Cycle*) in cluster 5 (left) and CLN2 (*CycLiN*) in cluster 1 (right). Red indicates the predicted values and black represents the original data.

in the absence of the observations. In the predicted pathways, it is important to note that there is a one-step delay in computing the effect of all prototypes on time-series microarray data (i.e. the prototypes determine the expression value of the cluster prototype 1 in the next time point). Thus the predicted curve is shifted with respect to the observations.

In the framework, we provided two different distance-based visualization approaches, direct mapping with cluster information and data positioning with multidimensional scaling. As she explores further, the user brings up these distance-based visualizations, which provide useful insights as described next.

Figure 20 shows two different visual layouts of gene expressions. Even though the genes are nominally different from each other, the visualization indicates that the genes in question have similar pathways or patterns. This is important knowledge for directing further investigations. In the left-hand case, we find there is a close-relation between the genes SEC2 (*SECretory*) and UNG1 (*Uracil DNA N-Glycosylase*). It is

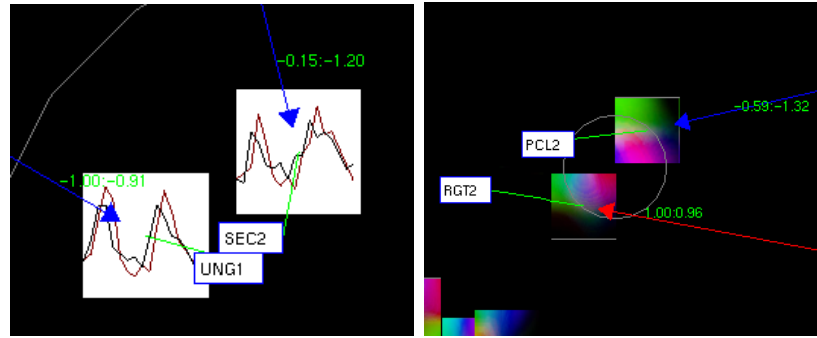


Figure 20: Similar gene expressions (left) and sequence information (right) are positioned close to each other in visual space.

known that SEC2 and UNG1 have the same upstream³ of the ATG start codon [151]. On the other hand, in the right-hand case the genes PCL2 (*PHO85 CycLin*) and RGT2 (*Restores Glucose Transport*) have different molecular functions even though they have similar sequences [131].

This example shows how interactive visualization supports rapid and efficient exploration of the data followed by zooming in on subtle but important similarities and differences. The user can then build on these insights by comparing with other regulatory behavior, looking closely at the annotated gene sequencing or protein expressions, as in Figures 17-18, or launching BLAST to bring up other annotated genes or proteins for comparative analysis.

Finally, we did some preliminary evaluations with bioinformaticists that found some strengths and weakness about the application. Strengths are (1) the capability of using both sequence analysis and pathway analysis in the microarray data, (2) the ability of discriminating the gene interactions in terms of local similarity and global similarity, and (3) predicting the pathway values in future time-stamps. Especially it

³Each strand of DNA or RNA has a 5' end and a 3' end. Upstream is the region towards the 5' end of the strand relative to the position on the strand.

has been noticed that sequence analysis while doing pathway analysis might usefully lead to further analysis on the gene itself. But further study has to be performed in determining the full use of sequence analysis in pathway analysis. Even though predicting pathway values is quite useful to drug discovery, fully understanding the concept of predictions in 3D and 2D visual layouts is difficult. Also 3D representation is useful to understand the concept of pathway analysis, but it has some limitation in the manipulating of the 3D visual layout with a 2D mouse.

4.1.1.9 Discussion

We designed an interactive analysis framework, with which the user can develop understanding of the dynamic regulatory pathways among genes by using visual analysis coupled with a prediction method. The framework implements a powerful integrated analysis that supports both understanding of the gene interactions over time and the understanding of gene function and structure (through comparative analysis). To support the analyzing procedures, we developed visual analyses in terms of two design steps, *Visual Representation* and *Visual Analysis*. In the framework, several interactive analysis features are provided: time-based cluster visualization, pixel-based visualization, simple line-graph layouts, multi-layered navigation tools, and sequence analyzing features. With these features, a user can quickly and effectively move among different perspectives to build an understanding of the time series structure, the gene interactions, their annotations, and their functional meanings. We have given a brief example of how these biological insights can be obtained for a particular gene regulation analysis. This integrated approach is quite important

because it supports what the analyst must do anyway and, up to now, has had to do laboriously without an integrated set of tools.

4.1.2 iPCA: An Interactive System for PCA-based Visual Analytics

Principle Component Analysis (PCA) is a widely used mathematical technique for high dimension data analysis. Just within the fields of computer graphics and visualization alone, PCA has been used in many different research areas [74]. At its core, PCA is a method that projects a dataset to a new coordinate system by determining the eigenvectors and eigenvalues of a matrix (Figure 21). This method finds the factors which explain the most variation among data points.

Although PCA is a powerful technique capable of reducing dimensions and revealing relationships among data items, it has traditionally been viewed as a “black box” approach that is difficult to grasp for many of its users [74, 122]. The process and result of the coordinate transform from original data space into eigenspace in PCA makes it challenging for the end user to identify the relationships between the input data and the data after the projection into eigenspace. This is especially problematic for novice users and students who need to use PCA but do not yet grasp how it works. Without a certain amount of background knowledge in the math behind PCA, it is often difficult for the user to perform effective analysis both in understanding how the original data items transform between coordinate systems and how the data dimensions relate to the principle components.

In order to assist the user in better understanding and utilizing PCA for analysis, we have developed a system called iPCA (interactive PCA) that visualizes the results

of principle component analysis using multiple coordinated views and a rich set of user interactions. The four coordinated views in our system visualize the data items in original data space (Data View), the data items in eigenspace (Eigenvector View), the data items projected onto two principle components (Projection View), and the correlations between all data dimensions (Correlation View). User interactions in one view are immediately reflected in the others so that the user can easily identify a data item or a data dimension in the original data space and its counterpart in eigenspace.

To demonstrate the effectiveness of iPCA, we performed a comparative user study with a well-known commercial system called Interactive Data Exploration, which is part of SAS/INSIGHT. The two systems are similar in that both systems use the same mathematical functions for performing PCA calculations, but they differ in their approaches to interface and interaction design. While the visualizations and interactions in our system are fluid, dynamic, and coordinated, in SAS/INSIGHT, a more traditional menu-driven and command-line approach forms the basis of interaction. Using SAS/INSIGHT, the user iteratively inputs parameters into the system before clicking on a button (or typing in a command) to initiate the PCA process and generate the results as static images and charts.

Our user study involved 12 participants performing complex analysis tasks on high dimensional data using both iPCA and SAS/INSIGHT. We quantitatively measured the accuracy and speed of the users' analyses, and asked the participants for qualitative feedback on ease of use, preference, and effectiveness. Based on the quantitative results of the user study, we find that users were faster and more accurate in analysis tasks using our system. Participants' feedback indicates that our system better facili-

tates the understanding of PCA, is more intuitive to use, and is unanimously preferred over SAS/INSIGHT. Many participants attributed the success of our system to its high interactivity and transparency, which suggests that our system is successful in opening up the “black box” of principle component analysis.

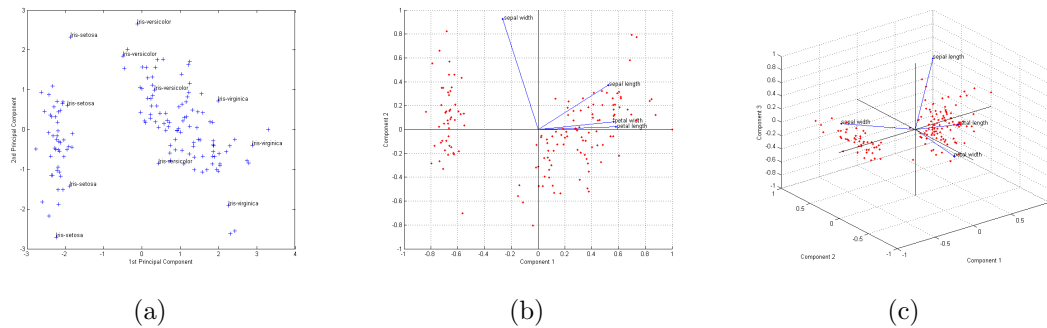


Figure 21: Examples of PCA with public Iris dataset [10]. A generic representation of data with the first two principal components (a), and 2D (b) and 3D (c) representations with principal components and the coefficients for each variable. The red dots in (b) and (c) indicate individual data items and blue lines represent the coefficients.

4.1.2.1 Related Work

PCA has been applied in many disciplines for various purposes. In visualization, PCA is used mostly for dimension reduction. For example, Hibbs et al. [64] apply PCA to visually analyze microarray data. Wall et al. [145] demonstrate how to visualize gene expression data using PCA and how to interpret the results. However, while PCA is popular and effective as a tool, there have been few available products or research projects on assisting the understanding of PCA results. Mathematical applications such as MATLAB [133] and SAS/INSIGHT [118] can perform PCA and visualize its results accurately. An open-source visualization tool, GGobi [132], supports interactive analysis of data through PCA and can be linked to R (Statistical Computing Software) for additional statistical methods. Müller and Alexa [93] de-

veloped a system which allows the user to visually detect and create clusters of data elements in the PCA space. Müller et al. [94] further enhanced conventional information visualizations with PCA and demonstrated that this combination improved data analysis. All these PCA-based tools are powerful and employ various visualization techniques. However, they also share the same goal of utilizing PCA with the assumption that users are experts at mentally transforming data elements from their original space into the projected PCA space. Our work differs in that we intend to use interaction to make the transformation of coordinate spaces intuitive to both novices and experts, and to show that by opening this “black box,” users can gain a deeper understanding of data analysis using PCA.

Interaction plays an important role in visualization for assisting users in understanding their data. Several user evaluations have found a benefit for interactive visual systems over traditional iterative input systems in understanding and using data. Ahlberg et al.[7] study the difference between using dynamic sliders and traditional text entry to visually explore periodic table data. They find that participants are faster with the dynamic slider interface on some but not all of their tasks. However, they do not find a clear difference in the participants’ subjective evaluation of the interfaces.

More recently, Callahan and Koenemann [23] compare an interactive visual tool, InfoZoom, against two traditional interfaces for online catalog browsing. With InfoZoom, users are more likely to complete tasks faster. The users also report higher ease of use and efficiency than traditional interfaces. In contrast, Combs and Bederson [32] compare a zoomable image browser to a static image browser and find no difference in

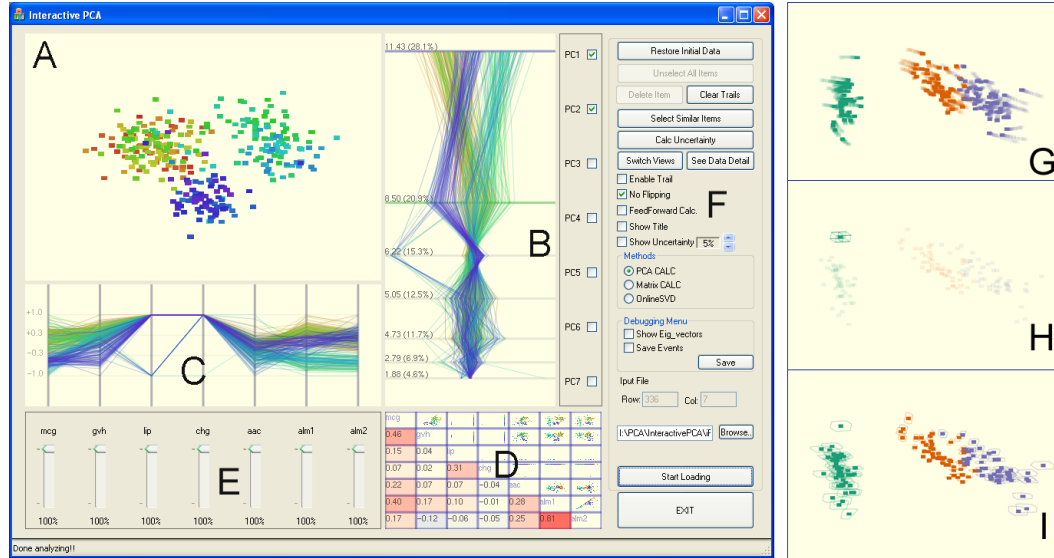


Figure 22: The system overview (left) showing the four views and the two control panels with the E. Coli dataset, and three examples with the Iris dataset (right). (A) Projection view. Data items are projected onto the two user-selected eigenvectors (in this case, the primary and secondary principle components). (B) Eigenvector view. Each eigenvector is treated as a dimension in this parallel coordinates view, and every data item is drawn as a line. (C) Data view. Another parallel coordinates view, but this time each dimension represents the dimensions in the original data, and each line represents each data item. (D) Correlation view. Pearson-correlation coefficient and the relationships (scatter plot) between each pair of variables are represented. (E) Dimension sliders. Each slider controls the amount of contribution of a dimension in the PCA calculation. (F) Control options. (G) shows the result of diminishing the first dimension (Sepal length) of the Iris dataset from 100% to 0%. The trails show how the data points move in PCA space in response to the change. The images (H) and (I) show 10% uncertainty in the data (in all dimensions). The possible locations for each data point are drawn in a hypercube (H) and in outlines (I) corresponding to the number of data item(s) selected.

performance, although users tend to (non-significantly) prefer the zoomable browser.

Unfortunately, while these studies inform us of the value of interaction, the tasks are simpler than asking users to perform complex analysis using PCA. Although some research studies [117, 121] have been performed to understand the effects of interactions in a more complex analysis task, these studies are narrowed to finding the effectiveness and the limitations of their applications.

4.1.2.2 Interface Design

The overall interface design of our system, iPCA, is based on multiple coordinated views. Each of the four views in the system represents a specific aspect of the input data either in data space or eigenspace, and are coordinated in such a way that any interaction with one view is immediately reflected in all the other views (brushing & linking). The coordination between the views depicts the same data item or data dimension in both data space and eigenspace simultaneously, thus allowing the user to infer the relationships between the two coordinate spaces.

Along with two control panels, iPCA contains four distinct views: the Projection View (Figure 22A), the Eigenvector View (Figure 22B), the Data View (Figure 22C), and the Correlation View (Figure 22D).

Projection View Two principal components (by default, the first and second most dominant eigenvectors) are used to project data points onto a two-dimensional coordinate system.

Data View The Data View is located below the Projection View, and shows a parallel coordinates visualization of all data points in the original data dimensions. In this view, an auto-scaling function is applied to increase the readability of data.

Eigenvector View In the Eigenvector View, data points are shown in the eigenspace. The calculated eigenvectors and their eigenvalues are displayed in a vertically projected parallel coordinates visualization, with eigenvectors ranked from top to bottom by dominance. The distances between eigenvectors in the parallel coordinate view vary based on their eigenvalues, separating the eigenvectors based on their mathe-

matical weights.

Correlation View Pearson-correlation coefficients and relationships between variables are represented in the Correlation View as a matrix of scatter plots and values. Since correlations between dimensions are symmetric, repetition is avoided by separating the matrix into three components: the diagonal, the bottom triangle, and the top triangle. The diagonal displays the name of the dimension as a text string. The bottom triangle shows the coefficient value between two dimensions with a color indicating positive (red), neutral (white), and negative (blue) correlations. The top triangle contains cells of scatter plots in which all data items are projected onto the two intersecting dimensions. The colors of the data items are the same as the colors used in the other three views so that clusters are easily identified.

It is relevant to note that the selection operation in all views and the zooming-in mechanism in the Projection and Correlation views help users to focus their interest on a data item or items. Also, the Projection View and the Correlation View can be switched such that the Projection View takes up the lower right hand position and the Correlation View fills the main display. This simple switch operation allows the user to utilize the visual real estate for focusing either on a single projection of data or to examine in detail all (or one) scatter plot(s) in the Correlation View.

The two control panels include a set of dimension sliders (Figure 22E) that can be used to decrease or increase the contributions of each of the original data dimensions, whose purpose will be discussed further in the following section (section 4.1.2.3). Several additional modes can also be specified in the other control panel to enhance

understanding of the visual changes during data analysis (Figure 22F). The user can enable *trails* so that the path of each data point’s recent motion is painted to the screen, making the movement of each point during interaction operations more apparent. The user can also choose to show *uncertainty* (Figure 22H and I) by setting a percentage of possible error in the dataset, which is reflected as bounding boxes around data items in the Projection View.

4.1.2.3 Interaction

Since iPCA is designed with high interactivity in mind, the types of available interactions are carefully considered. We categorize all the interactions in iPCA into two groups: interactions with the views, and interactions with PCA. Interactions with the views are operations that do not result in PCA calculations, and include brushing, filtering, zooming and panning, etc; whereas interactions with PCA will result in new PCA calculations, including operations that change the weights of dimensions, move data points in either data space and eigenspace, and removal of data points. Both types of interactions are embedded in the coordinated views such that all views react to all interactions.

4.1.2.3.1 Interacting with the Views

Interactions in this category are operations that do not cause the system to re-compute PCA. As mentioned above, these operations include brushing, filtering of data items or dimensions, zooming and panning, etc. Although these interactions are standard in most Infovis or visual analytics tools, they are nonetheless very important, and are essential in multiple coordinated views. The ability to allow the user to

select a cluster of data items in one coordinate space and immediately see the corresponding items highlighted in the other coordinate space helps the user understand the relationship between the two.

The most notable interactions in this category are the different types of selections implemented in iPCA. iPCA allows the user to select data items in all four views. In Data View and Eigenvector View, where the visualizations are parallel coordinates, selection means clicking on a single line or brushing a range of items. In Projection View and Correlation View, the user can either click on a single dot or draw an enclosed space upon which all data items within the space will be selected.

4.1.2.3.2 Interacting with PCA

As mentioned previously, one of the biggest hurdles in effectively analyzing PCA results is in understanding the relationships between data space and eigenspace. While the interactions provided in the previous section allow the user to see a data item appear in different coordinate systems, the interactions do not immediately lead the user to see the relationship between the coordinate spaces. Specifically, eigenvectors are linear combinations of data dimensions, therefore, understanding which data dimension contributes to an eigenvector is a key point in comprehending how the coordinate spaces relate to each other.

In order to visually assist the user in recognizing how data space relates to eigenspace, we create a set of interactions that allow the user to alter the values of the data items. For example, if the user drags a data item in the Projection View towards the positive direction along the x -axis (increasing the data points value in the first principle com-

ponent), the user should be able to immediately observe in the Data View how that change affects the values of that data item in the original data space, thus shedding light on the relationship between the first principle component and all dimensions in the original data space.

Similarly, if there is an obvious cluster in the Projection View, the user can interactively change the weights of a dimension to see its affect on the formation of the cluster. For example, if diminishing the contribution of a data dimension in PCA calculation down to 0% does not affect the clustering, then it should be clear that the cluster does not depend on that particular dimension.

While the concept of encouraging interactions that directly alter the values of data items seem counter-intuitive, the idea is not novel. *Spotfire* includes a “jitter” operation [6], and *Dust and Magnet* has a “dust shake” operation [158], both of which are designed to reveal occluded data items. In medical visualization, deformation or “cut-aways” modify the data to expose hidden structures underneath skin and flesh [89]. The interactions in iPCA share a similar goal, but instead of revealing hidden or occluded information, our interactions assist the user in revealing relationships between coordinate spaces.

Three specific interactions are implemented based on the concept of data alteration: modifying dimension contribution, adjusting data items, and removal of data items.

Modifying Dimension Contribution Each slider in Figure 22E corresponds to a data dimension. By modifying the slider, the user can change the contribution of the data dimensions in the final PCA calculation. For instance, changing the dimension

contribution to 50% indicates the weight change of the selected dimension to 0.5. This interaction allows the user to observe which data dimensions contribute to the projections of the data in eigenspace. By adjusting these sliders, a user can quickly test hypotheses about how the analysis would be affected if a dimension or set of dimensions were removed or considered less important. This makes it possible for a user to observe the formation and dispersion of clusters and to identify the cause of outliers.

Adjust Data Items Values of data items can be modified in either the Projection View, Data View, or Eigenvector View. This interaction not only allows the user to see the relationship between a principle component and the contributing data dimensions as mentioned above, but also allows the user to test what-if scenarios. If the user suspects that a data item should appear in a certain cluster, the user can manually move the data item and see how the values of that data item would have to be modified.

Removing Data Items In analysis using PCA, a common task is for the user to remove outliers. iPCA supports direct removal of data items from the system so that the user can observe how the projection from data space to eigenspace changes with the removal.

One caveat of these interactions is that they are computationally expensive. Modifying any data requires the re-computation of PCA, and in the cases of interactively adjusting sliders and moving data items on screen, PCA has to be re-calculated quickly to avoid lag or flickering. For very large datasets, this type of interactions

has the potential of becoming a bottleneck in usability. In iPCA, the scalability issue is addressed by incorporating a faster version of singular value decomposition called online-SVD [15] which trades precision for speed. Brand demonstrates how online-SVD is faster than traditional SVD (see [15] for detail). The user has the option to use either traditional SVD or online-SVD depending on the speed and accuracy requirements as well as the scale of the data.

4.1.2.4 Evaluation

We conducted a comparative evaluation to assess the effectiveness of our system in relation to a well-known commercial tool, SAS/INSIGHT’s Interactive Data Exploration. A total of 12 students (nine males) participated in the evaluation. Three of the 12 participants were undergraduate students and nine were graduate students, and 11 of the participants majored in computer science and one in management of information science. Based on self reported familiarity, we found that nine participants were aware of PCA prior to the evaluation, and of the nine, three had used PCA in the past.

At the start of the evaluation, all participants receive a detailed explanation about PCA followed by a pre-evaluation background questionnaire. Each participant was provided a total of ten minutes to train with the two systems prior to the evaluation. The evaluation consisted of performing four analysis tasks using each system. The participants were given five minutes to perform each task and were requested to answer questions immediately after each task. The evaluation was conducted using an online website, where time spent and answers were saved into a database.

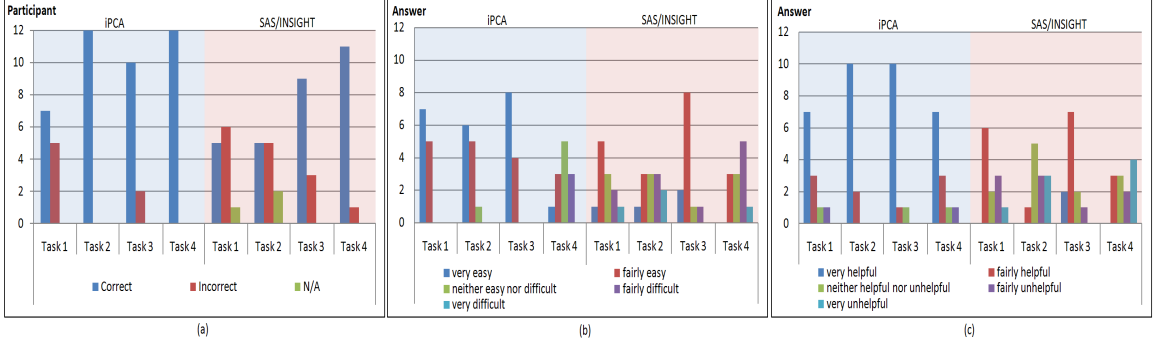


Figure 23: Results broken down by tasks for each of the two systems. (a) Number of participants who answered the task question correctly. (b) Task difficulty and (c) helpfulness of the system in solving the task, as reported by participants.

We performed the evaluation using three different datasets: the Iris dataset (150 data items \times 4 dimensions), the E.Coli dataset (336 data items \times 7 dimensions) and the Wine dataset (179 data items \times 13 dimensions). The Iris dataset was used in the training session whereas the E.Coli and the Wine datasets were used in the actual evaluation. All three datasets are scientific results that are publicly available at the UCI Machine Learning Repository [10].

4.1.2.4.1 Procedure

Each participant was requested to use the two systems on different datasets. Therefore, six participants used iPCA first and the rest of the participants began with SAS/INSIGHT. The order in which datasets were given to each participant was counterbalanced with system order, so that six participants used the E.Coli dataset first and the rest used the Wine dataset first.

Four tasks were given to each participant during the evaluation of both systems:

- What is the most striking outlier you can find? An outlier is a point that does not fit the overall patterns of the dataset.

- Find a dimension that least affects the PCA outputs in the Projection View using first and second principle components.
- Find two dimensions with a highly positive correlation. Also find the class name and label of an outlier that does not follow that correlation.
- How does removing the first dimension affect the PCA results using the first and second principle components? List as many observations as possible.

The first three tasks are related to finding exact answers and the last one is a descriptive task asking the participant to describe the difference between including and excluding a specific dimension. Five minutes were given to solve each task. If time expired, partial answers were saved into the database. As soon as each task was completed, a post-task questionnaire was given to participants to track how they felt about the task. These questions included “How difficult was this task?” and “How helpful was the interface in solving the task?” A post-application questionnaire was given after a participant completed all four tasks. This questionnaire asked the participant to give feedback on their overall subjective opinion about each system. After a participant completed the evaluation using both systems, the participant completed an additional set of questions (post-study questionnaire) that described the preference, the ease of use, and the effectiveness of the system in analyzing data. Finally, the participant graded each system on a scale of ‘A’ to ‘F’.

4.1.2.4.2 Evaluation Results

We present the results of our evaluation based on accuracy, speed, difficulty and usefulness, effectiveness, and preference. Both accuracy and speed are measured quantitatively; whereas the other three categories are analyzed based on the participants' qualitative feedback.

Accuracy Figure 23(a) shows the results of the participants' accuracy in solving each task using both iPCA and SAS/INSIGHT. As shown, approximately 85% of the participants answered correctly using iPCA. On the other hand, when using SAS/INSIGHT, they were only able to answer correctly 62% of the time. Furthermore, when using SAS/INSIGHT, there were three instances in which a participant could not complete the task. One of the instances was due to the fact that the participant ran out of time. In the other two cases, the participants simply gave up and claimed that they were unable to find the solutions (see Figure 23(a)). Note that the accuracy difference is statistically significant across the two systems ($p < 0.01$).

Speed Table 1 shows the overall average time spent solving each task. Participants spent less time in solving each task using iPCA except for task 4 (a descriptive

Table 1: Average time spent in solving each task.

Application	Task	Time Spent (seconds)
iPCA	Task 1	136.58 \pm 90.91
	Task 2	128.33 \pm 93.83
	Task 3	125.50 \pm 50.57
	Task 4	211.33 \pm 71.15
SAS/INSIGHT	Task 1	177.67 \pm 69.74
	Task 2	165.58 \pm 84.03
	Task 3	142.92 \pm 66.19
	Task 4	197.08 \pm 63.61

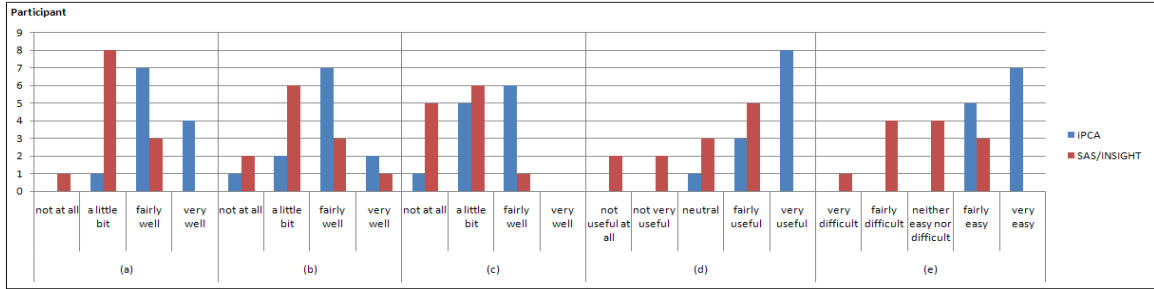


Figure 24: Participants' responses to a post-application questionnaire, filled out after solving all four tasks using one of the systems. (a) How well do you understand the application now? (b) How well do you understand PCA now? (c) How well do you understand the data you worked with now? (d) How useful was the system? (e) How difficult or easy was the system to learn?

question). This seems to be because participants tried to find as many differences as possible through interaction with dimensions.

On average, participants spent about 150 seconds using iPCA, and 170 seconds using SAS/INSIGHT. Although the difference is not statistically significant ($p = 0.17$), there is a trending effect towards a faster solution when using iPCA.

Difficulty & usefulness (post-task questionnaire) Figure 23(b) and (c) show how participants rated the difficulty of each task when using iPCA or SAS/INSIGHT, as well as how they rated the usefulness of each system in solving the task. Figure 23(b) indicates that about 81% of the participants found the tasks to be easy when using iPCA. On the other hand, only about 48% of the participants identified the tasks as being easy when using SAS/INSIGHT. Interestingly, although more than half of the participants mentioned that task 1 is easy to solve, Figure 23(a) indicates that the accuracy in solving task 1 is low (58% iPCA and 41% SAS/INSIGHT). This might be because most participants have little previous experience with finding outliers.

Figure 23(c) shows that about 90% of the participants identified iPCA to be helpful

in solving the tasks; whereas only about 40% of the participants found SAS/INSIGHT to be helpful. Furthermore, only two participants (participant D and I) rated iPCA to be not helpful in solving a task (tasks 4 and 1, respectively); whereas ten participants indicated that SAS/INSIGHT was unhelpful in solving some tasks (one participant indicated SAS/INSIGHT was completely not helpful in solving all tasks).

Overall, we find that the more difficult a task was rated (very easy = 5, fairly easy = 4, etc), the more time the participants spent on solving it ($p < 0.0001$). However, solving a task with a “helpful” system (very helpful = 5, fairly helpful = 4, etc) did not decrease the time spent on the task ($p = 0.2586$). With a helpful system, the participants did solve the tasks more accurately ($p = 0.0027$), but participants did not rate the tasks to be less difficult ($p = 0.0966$).

Effectiveness (post-application questionnaire) Figure 24 shows the results of the five questions in the post-application questionnaire conducted right after the evaluation of each system. Of particular significance are the questions asking the participants how well they understood the application (Figure 24(a)), how well they understood the data (Figure 24(c)), how useful was the system (Figure 24(d)), and how difficult or easy was the system to learn (Figure 24(e)).

In answering how well the participants understood PCA, most participants did not indicate that they understood PCA “very well.” However, the majority of the iPCA users indicated that they understood PCA “fairly well”; whereas the majority of the SAS/INSIGHT users only claimed “a little bit” of understanding.

In answering how well the participants understood the data, the majority of the

iPCA users indicated that they understood the data either “fairly well” or “a little bit”; whereas the SAS/INSIGHT users either understood the data “a little bit” or “not at all.”

Lastly, in answering about the usefulness of the system, the majority of the iPCA users found the system to be “very useful”; whereas SAS/INSIGHT users consistently ranked the system to be “fairly useful” and below, with four participants claiming the system to be “not very useful” or “not useful at all.”

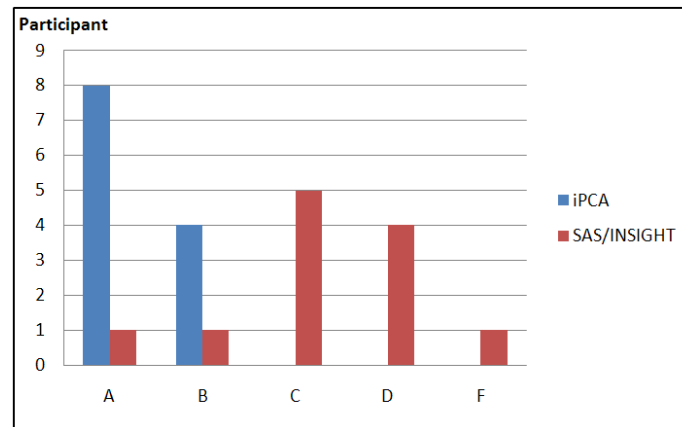


Figure 25: At the end of the evaluation, each participant grades the systems on a scale of ‘A’ to ‘F’.

Preference (post-study questionnaire) After the evaluation, each participant ranked the two systems and described their pros and cons. Figure 25 clearly shows that most participants preferred iPCA over SAS/INSIGHT, giving iPCA eight A’s and four B’s. On the other hand, the majority of the participants gave SAS/INSIGHT a C or D grade, with one participant failing it by giving it an F.

When describing the pros and cons of iPCA, eight participants specifically pointed out the strength of iPCA as being “interactive,” and eight participants described iPCA as “transparent.” While a few participants gave constructive feedback on how

to further enhance the iPCA tool (e.g., add the ability to rearrange the dimensions in the Correlation View), the only negative criticism for iPCA was that it did not generate printable reports similar to the static charts and numbers that SAS/INSIGHT generates.

For SAS/INSIGHT, two participants who were previously familiar with the SAS system pointed out that while they preferred iPCA over SAS/INSIGHT for analyzing PCA results, SAS is still a far more comprehensive and complete numerical and statistical analysis tool. One participant further noted that if he was allowed to use additional features in SAS outside of the tools specific to PCA, deeper analysis on the dataset could have been performed.

4.1.2.5 Discussion

Since our evaluation compared two systems (iPCA and SAS/INSIGHT) that use the same mathematical methods for computing PCA, we can safely assume that the increase in our participants' performance in using iPCA is attributed solely to the interface design and the set of interactions. Unfortunately, we are not able to further isolate the specific factor(s). Based on our evaluation alone, we cannot determine if the increase is due to the multiple coordinated views, the interactions, or the combination of the two. However, we do hypothesize that the "interactions with PCA" play a significant role in that the user's direct and continuous manipulation with PCA is rewarded with immediate visual feedback. This allows the user to "play" with the data and intuit the subtleties behind the coordinate transform between data space and eigenspace in a way that less interactive visualizations such as SAS/INSIGHT

cannot achieve.

The “interactions with PCA” are also the most unique set of the interactions in iPCA. Unlike our other interactions that merely highlight or explore the data, the design decision behind the “interactions with PCA” is to focus on reasoning. In fact, we design the “interactions with PCA” to be less faithful to the data, but more revealing in discovering relationships between coordinate spaces and data dimensions. For example, most of our participants credited the rich interactions in iPCA to be the primary strength of the system, but two of our participants pointed out the fact that in modifying dimension contribution, moving a slider from 100% to 72% and taking a snapshot of the Projection View was not meaningful as the projection was not of the original data. Similarly, moving a data point across the screen seemed counter-intuitive as it directly modified the values of the data. While these concerns are valid, we contend that they miss the spirit of the interactions. It is true that the resulting images from these interactions cannot be considered by themselves, but it is during the direct manipulation of the data and coordinate spaces that the user gains insight about their relations and how changes in one affects the other, which is otherwise hidden. One very interesting future direction for our research will be to further understand why these types of interactions are successful, and examine the extent to which they can be applied.

4.2 Externalization

Externalization is the process of articulating tacit knowledge into explicit concepts [97]. It is a generic knowledge creation process through which tacit knowledge

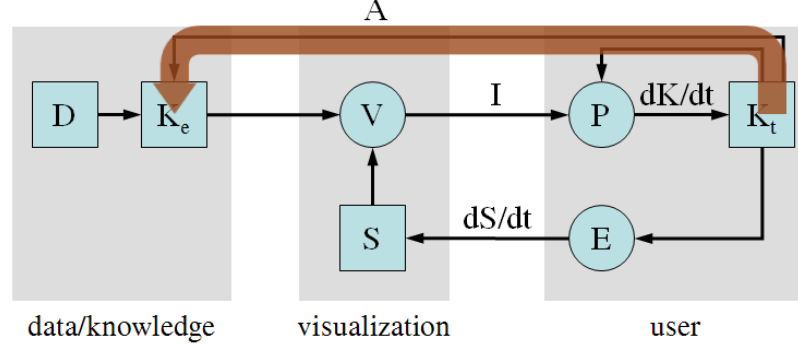


Figure 26: Externalization process (indicated by the red arrow). It shows that the internally created tacit knowledge can be extracted and saved into the knowledge base. The explicit knowledge can later be reused to create further visual representations.

becomes explicit based on the user's finding (insights), concepts, hypotheses, and models. Since tacit knowledge cannot easily be shared with others as a result of simply being written down, it should be converted to explicit knowledge by an externalization process beforehand to communicate and share with others.

In the visualization community, there have been a few applications that specifically focus on the externalization of one's tacit knowledge. Garg et al. [56] presented a model-driven approach to extract Logic Programming (LP) rules through a user's interactions and reused the rules in further analysis processes. Xiao et al. [155] studied how the knowledge-base could be used to improve understanding of complex network traffic data. They found that about 80% of network traffic could be classified correctly based on previously extracted experts' knowledge-base. Chen et al. [27] showed an example in which the user's insights can be externalized into a knowledge base. These applications have shown that not only is externalization in a visualization possible, it is in fact a very powerful method for storing and reusing knowledge.

Figure 26 shows how tacit knowledge can be externalized into explicit knowledge. In

this figure, explicit knowledge is used to assist the creations of visual representations. Because of the complex nature of explicit knowledge, Nonaka and Takeuchi suggest that externalization is the process of concept creation and is triggered by dialogue or collective reflection [97]. They further explain that the concept creation process can be expressed by applying analytical and non-analytical methods alike. The analytical methods are deduction and induction through which appropriate concept can be deduced and induced based on unorganized concepts. However, if applying analytical methods is not feasible, non-analytical methods such as metaphors and/or analogies could also be used. In visualization, we propose that both analytical and non-analytical methods should be considered for expressing explicit knowledge that can in turn be stored into a knowledge base [155, 56].

As we show in section 4.1, interactive visualization techniques enable users to more effectively form hypotheses, identify areas for more detailed investigation, and to build tacit knowledge. However, there has been limited effort in connecting a user's interaction with his reasoning for the purpose of extracting the relationship between the two. In section 4.2.1, we present an approach for capturing and analyzing user interactions in a financial visual analytical tool and describe an exploratory user study that examines these interaction strategies. To achieve this goal, we created two visual tools to analyze raw interaction data captured during the user session. With the tools, we performed a study in order to understand the analysts' reasoning process for solving complex problems. In our study, we record the interactions of 10 financial analysts in a fraud detection task. By examining their interactions, we are able to quantitatively show that up to 60.10% of strategies, 60.08% of methods,

and 79.03% of findings could be recovered through the use of a strategic and an operational visual analytical tool. From the study, we found that users' interaction begets their reasoning.

In section 4.2.2, we present a system called Voting Viz. With this system, analysts can perform interactive exploration on wire transactions, vote on transaction(s) as suspicious, unsuspicious, and inconclusive, and examine other analysts' voting information. Since analysts use their background knowledge (experience) to find suspicious activities on wire transactions, all their decisions could include their personalized knowledge. To integrate their knowledge into visualization, a voting method is designed. With the voting method, each analyst can make decisions on transactions as suspicious, unsuspicious, or inconclusive. In addition, an annotation method is provided to document the analyst's reasoning process.

From the two examples, we demonstrate how users continuously create personalized tacit knowledge by using visualization or visual analytics applications, and that the users' tacit knowledge can be externalized and used to design a new visualization representation.

4.2.1 Understanding experts' reasoning process from interactions

In this section, we performed a study to show how an analyst performs a successful investigation with a visual analytics system. Although there has been a recent increase in activity in the VA community to help analysts document and communicate their reasoning processes during investigations, there is still no clear method for capturing the reasoning processes with minimal cognitive effort from the analyst. We seek to

determine how much an analyst’s strategy, methods, and findings using a VA tool can be recovered.

We hypothesize that when an analyst interacts with a well-designed VA tool, much of that analyst’s reasoning process is embedded within his or her interactions with the tool. Therefore, through careful examination of the analyst’s interaction logs, we should be able to retrieve a great deal of the analyst’s reasoning process. To validate our hypothesis, we designed a study to quantitatively measure whether an analyst’s strategies, methods, and findings can be recovered through human examination of these interaction logs.

4.2.1.1 Related Work

Although these publications have different emphases and different approaches, most of them aim to measure and validate the effectiveness of visualization tools and visual analytics as a science. Among this growing literature, the areas most relevant to our work are efforts in capturing analysts’ reasoning processes into two groups: capturing the user’s interactions and interactive construction of the reasoning process using a visual tool.

4.2.1.1.1 Capturing User Interactions

Capturing user interactions for the purpose of understanding the user’s behavior is common in both academics and industry. Some commercial-off-the-shelf applications capture a user’s desktop activities (such as usability software), whereas others capture interactions on a Web site (a common feature in most Web servers).

In the visualization field, Frank Greitzer’s Glass Box system is one of the most

notable systems for capturing and analyzing user activities [59]. Its primary goal is to capture, archive, and retrieve user interactions [33]. However, it is also an effective tool for capturing specific types of interactions for the purpose of intelligence analysis [34]. Although Glass Box and most usability software are effective tools for capturing user activities, they focus primarily on low-level events (such as copy, paste, mouse clicks, and window activation). The events captured in our system are at a higher level that corresponds directly to the data (such as the transaction the user clicked on). More information on the differences between these two approaches is available elsewhere [63].

More recently, Jankun-Kelly et al. proposed a comprehensive model for capturing user interactions within a visualization tool [70]. Their work is unique in that they focus on capturing the effects of the interactions on the parameters of a visualization. Although it is unclear how this framework supports higher-level event capturing, the direction is interesting and could lead to a more uniform way of capturing user interactions.

These systems and approaches are all innovative and effective. However, their objectives differ from our goal in that none of them fully address how much of a reasoning process can be recovered through the examination of interaction logs.

4.2.1.1.2 Interactive Construction of the Reasoning Process

An alternative approach to retrieving reasoning through interactions is for the analyst to create a representation of the reasoning process (usually in the form of a node-link diagram) while solving a complex task. Recent systems in this domain

include the Aruvi framework [123], which contains three main views:

- The data view is the VA tool itself.
- The navigation view is a panel for visually tracking the user’s history.
- The knowledge view lets the user interactively record his or her reasoning process through the creation of a node-link diagram.

The Scalable Reasoning System (SRS) also lets users record their reasoning processes through the creation of node-link diagrams [103]. However, unlike the Aruvi framework, SRS focuses on the collaborative aspects of organizing the reasoning processes among multiple users and sharing their results across the Web.

Most recently, Jeffrey Heer and his colleagues⁴ created a tool for visualizing users histories within the commercial visualization tool Tableau [86]. Although the work’s emphasis is not on constructing or visualizing the reasoning process, the functionalities within the tool that let users edit and modify their interaction history could be used to effectively communicate their reasoning processes.

Although there has not been a formal comparison between interactively constructing the reasoning process and our method of analyzing interaction logs, we hypothesize that the cognitive load of having to perform analytical tasks while maintaining and updating a representation of the reasoning process could be tiring [58]. The systems we describe here will have better representations of the user’s reasoning process. However, a transparent, postanalysis approach offers an alternative that can achieve comparable results without the analysts’ efforts. The best solution is most likely

somewhere in between, and we look forward to analyzing the pros and cons of these approaches.

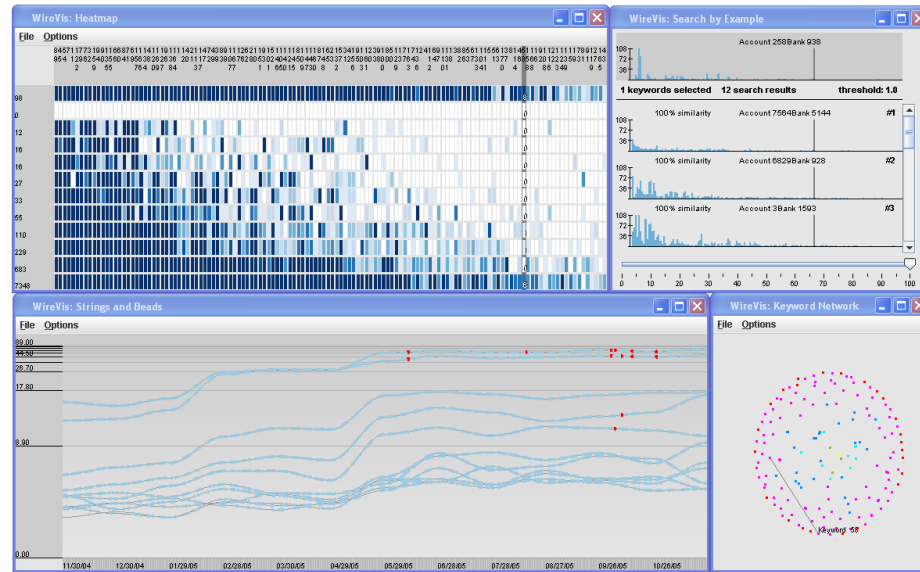


Figure 27: Overview of WireVis. It consists of four views including a Heatmap view (top left), a Strings and Beads view (bottom left), a Search by Example view (top right), and a Keyword Relation View (bottom right) [25].

4.2.1.2 WireVis Interactions

We conducted our study using WireVis, a hierarchical, interactive VA tool that logs all user interactions. WireVis was developed jointly with wire analysts at Bank of America for discovering suspicious wire transactions. It is currently installed at Bank of America’s wire-monitoring group, WireWatch, for beta testing. Although it has not been officially deployed, WireVis has already revealed aspects of wire activities that analysts were not previously able to analyze. Through a multiview approach, the tool depicts the relationships among accounts, time, and transaction keywords within wire transactions (see Figure 27).

To preserve the privacy of Bank of America and its individual account holders,

we created a synthetic data set for the purpose of this study. Although none of the transactions in the data set are real, we captured as many characteristics and statistics from real financial transactions as we could and modeled the synthetic data as closely to the real one as possible. The data set was designed to be simple enough that users could look for suspicious transactions within the study’s time frame, but complex enough that interesting and complicated patterns could be found. This data set contained 300 financial transactions, with 29 keywords. Some keywords were the names of countries, such as Mexico, and others were goods or services, such as software or raw minerals.

In the data set, we developed four threat scenarios and injected a total of nine cases we deemed suspicious into the data set. The threat scenarios included transactions in which keywords should not appear together, accounts with dual roles, keywords with unusually high transaction amounts, and accounts with suspicious transactional patterns appearing over time. More details about the sample threat scenarios are:

- **Incompatible Keywords in a Transaction:** Transactions with two or more keywords that do not belong together. For example, a transaction containing the keywords “car parts” and “baby food”.
- **Accounts with Dual Roles:** An account that has had transactions of different incompatible keywords is questionable. For example, an account that transacts on “gems” at one time and “pharmaceuticals” at another.
- **Keywords with Large Amounts:** Transactions of certain keywords are expected to have corresponding dollar amounts. For example, a transactions from a local

store on “arts and crafts” should not be in the millions.

- **Change in Amounts Over Time:** An account with an established temporal and amounts pattern receiving a large sum outside of its norm should be examined further. For example, an account with a steady deposit of paychecks of fixed amounts on regular intervals receiving a transaction of a large amount.

We also developed two tools for visualizing user interactions within WireVis to help us explore analysts’ activities and reasoning processes [33]. Using the *operation analysis tool*, we can analyze a participant’s interactions in accordance to his reasoning process; while using the *strategy analysis tool*, we can compare the strategies of different participants or groups of participants. Together, these two tools provide the means to discover the relationship between user interactions and reasoning processes at all levels.

4.2.1.2.1 Operation Analysis Tool

The operation analysis tool is designed to support the analysis of a participant’s operational interactions in relation to his annotations (Figure 28). The tool is implemented in OpenGL, and is fully zoomable and pannable and supports selections of interaction elements for detailed inspection. The x -axis of the main view represents time, with a striped background indicating the length of a fixed time duration (defaulted to 60 seconds per strip). The y -axis is divided into 5 sections, with each section supporting one aspect of the participant’s investigation process. Figure 28 (A)-(E) show the 5 perspectives, which are the participant’s annotations (A), the participant’s interactions with the three views in WireVis (B), the depths of a par-

participant's investigation (C), the areas of the participant's investigation (D), and the time range of the investigation (E). The sliders in (F) allow the user to scale time, while checkboxes in (G) control various visualization parameters. The detail view in (H) depicts detailed information of a specific user-interaction element.

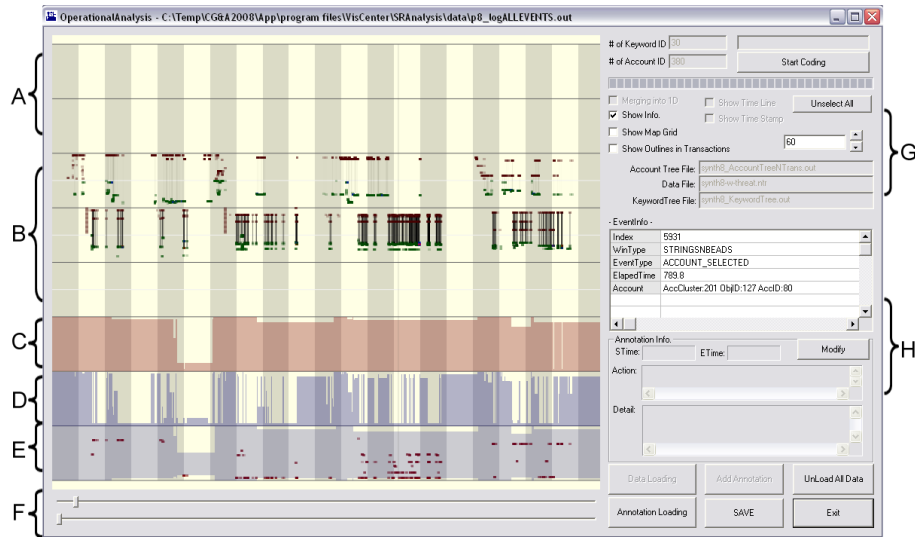


Figure 28: Overview of the operational-analysis tool: (a) a potential area for adding annotations; (b) the participant's interactions with the three views in WireVis (the three rows, from top to bottom, correspond to the Heatmap, String & Beads, and search-by-example views). Other highlighted areas include the (c) depths of a participant's investigation, (d) areas of the participant's investigation, and (e) time range. In addition, (f) sliders control the time scale, while (g) checkboxes change various visualization parameters. Another view (h) shows detailed information about the participant's selected interaction element.

Annotation View The results of our participants' think-aloud during the experiment are recorded into separate files. These annotations to the participant's investigation are shown in this view. As can be seen, our participants often exhibit a hierarchical structure in their reasoning process, with the highest level of reasoning depicting strategies they employ such as “seek all keywords related to the keyword food” The lower levels depict specific operations to execute those strategies, ranging from “search

for keywords other than food relating to account 154” to “identify the receiver of a transaction (account 64) of account 154.” The hierarchical nature of the participants’ reasoning are represented in the annotation view, with the higher level annotations shown above the lower ones as interactive floating text boxes [65]. The time range of each annotation is drawn as nested boxes using different colors. The user can select any particular annotation, and its corresponding time range is highlighted across all the other views (Figure 28).

WireVis Interaction View WireVis uses multiple coordinated views to visualize different relationships within the data. In the WireVis Interaction view, we look to display the participant’s usage pattern of the WireVis tool. The three rows in this view correspond to the three main interactive views in WireVis: Heatmap, Strings and Beads, and Search by Example. In each view, we can choose two different attributes of the participant’s selection. In Figure 28 (B), the two attributes are keywords (shown as red dots) and accounts (shown as green dots).

On first glance, it is easy to see which views in WireVis the participant interacts with over time. On closer inspection, the distribution of the red (keywords) and green dots (account) also reveal high-level patterns in the investigation. Scattered red dots could indicate an exploration of keywords, whereas concentrated green dots (e.g., if the green dots are aligned horizontally) could reveal the participant’s interest in a specific account. When both red and green dots appear together and are connected by a line, it denotes that the participant is investigating the relationship between the two (such as a cell in the Heatmap view in WireVis).

Depth View On top of visualizing a participant’s direct interactions with WireVis, it is also important to see some semantic information regarding the participant’s investigation process. In this view, we visualize the “depth” of a participant’s investigation by displaying the number of visible transactions in WireVis. For example, when the participant is looking at the overview in WireVis, our Depth view will be completely filled, indicating that all transactions are visible. As the participant zooms in to investigate specific keywords or accounts, the Depth view will show a drop in visible transactions (Figure 28 (C)).

The Depth view also indicates when a participant requests detailed information for a specific account or transaction (such as double-clicking on a bead in the Strings and Beads view). These interactions show up as a vertical line, which is easily distinguishable from a participant’s operations for zooming in or focusing on a specific area in the data.

Areas View While the Depth view shows the number of visible transactions in WireVis, it is also relevant to indicate interactions that highlight areas that the user has shown interests. These interactions are commonly used in WireVis through the “mousingover” operation. As the participant mouses-over keywords, accounts, or transactions in WireVis, the system displays information about the highlighted data without requiring the user to change the zoom level or focus.

Using the mouse-over operation in WireVis is common, and often indicates an exploration process in which the participant is looking for suspicious areas for further investigation. In the Areas view, a high variation in a short amount of time could

indicate such an exploration process, while a more leveled section suggests that the participant is investigating specific activities (Figure 28 (D)).

Time Range View Time is an important aspect in discovering financial fraud, and WireVis provides views to explore the temporal dimension. In the Time Range view, we look to capture the participant’s time-based investigation. The y -axis of the Time Range view denotes the dates represented in the data from more recent to least. A fully colored section indicates that the participant’s investigation spans the entire time range, whereas a change would denote that the participant has zoomed in to a specific time period (Figure 28) (E).

The dots in the Time Range view indicate selections of transactions of a specific date. In WireVis, this is done by either mousingover or double-clicking on a bead in the Strings and Beads view. A high concentration of the appearance of these dots often suggests that the participant has found some specific transactions and is looking to find out the details of these transactions.

4.2.1.2.2 Strategy Analysis Tool

As opposed to operation, strategy is a long term plan of action designed to achieve a particular goal. Most of our participants exhibit both strategic and operational reasoning when investigating fraud. So besides addressing the question “what do the participants actually do” using our operation analysis approach, we also look to investigate the high level strategies that the participants employ while approaching the tasks. Through the use of our strategy analysis tool, we can identify each participant’s areas of interest as well as comparing different participants’ strategies.

We adopt treemap as the basis of our strategy analysis tool. The treemap visualization allows us to investigate similarities between our participants' strategies without considering the flow or speed in which our participants execute their strategies. Our participants had varied preconceived knowledge about the keywords and their meanings, and therefore approached the investigation tasks differently. Many of them identified the same embedded fraud scenarios, but none of them shared the same path in discovering these activities. Using our modified treemap visualization, we can identify the participants' strategies without regard to the paths they have chosen.

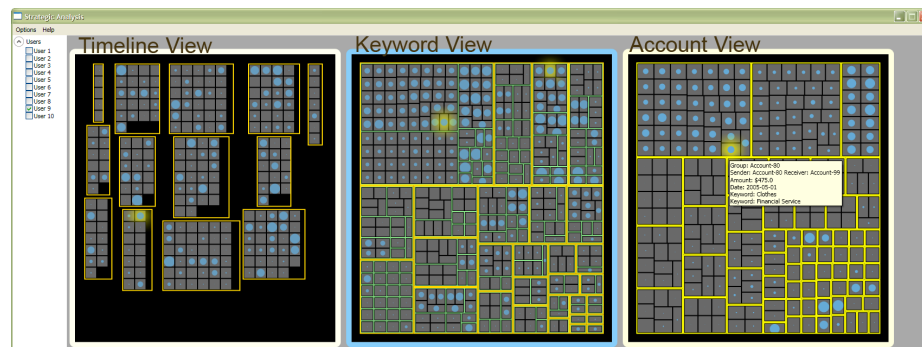


Figure 29: Strategic-analysis tool views: transactions grouped by (a) time, (b) keywords, and (c) accounts. The patterns in the account view indicate that the primary strategy used by this participant was to examine two specific accounts (located on the top of the view).

The initial layout of the strategy analysis tool shows three different treemap views classifying the transaction data based on three attributes: time, keywords, and accounts (Figure 29). The three views are coordinated such that highlighting a transaction in one view also highlights the same transaction in the other two views. We choose transactions to represent the lowest level of the treemaps because they represent the lowest granularity of the data. A colored circle is displayed on each cell,

and the size depicts the amount of time the participant’s investigation has included that transaction. When comparing two participants or two groups of participants, the color of the circle indicates which of the two participants spent more time on the transaction.

Timeline View Transactions in this view are classified based by their date. Each grouping contains transactions of the same month. As shown in Figure 29, the transactions in our synthetic data set span a 13 month period. Note that the participant depicted in this view did not perform his investigation based on the transaction dates as the circles appear fairly evenly through all 13 months.

Keyword View This view applies two different classification criteria. On the top level, transactions are grouped based on keywords (shown as yellow cells in Figure 29). Each cell is then further subdivided by individual accounts (shown as green cells).

Since a transaction often contains multiple keywords, the same transaction could appear in more than one keyword cells. Similarly, every transaction contains two accounts, a sender and a receiver, so a transaction will always appear at least twice, once for each account. Due to these two reasons, the total number of transaction cells in this view are greater than those in the Time view. However, we find this layout more intuitive for understanding a user’s strategy involving keywords. For example, in Figure 29, it is easy to see that the participant focused on a few specific keywords, but even more specifically on a few accounts relating to those keywords.

Account View The Account View orders the transactions based on their corresponding sending accounts. As shown in Figure 29, this view makes clear that this partici-

pant’s strategy in discovering financial fraud using WireVis is almost entirely based on the detailed investigation of one or two accounts. Time and keywords appear to be secondary considerations during his investigation.

4.2.1.3 Evaluation

We conducted a user study to determine how much of an analyst’s reasoning process can be recovered using only the captured user interactions. We evaluated this recovery quantitatively by comparing the process that was inferred by a set of coders against the ground truth determined from videos of the exploration process. The evaluation consisted of four stages: user observation, transcribing, coding, and grading.

4.2.1.3.1 User Observation

To understand the user’s reasoning process through his or her interactions, we conducted a qualitative, observational study of users analyzing data with WireVis. We recruited 10 financial analysts with an average of 9.9 years (and a median of 8 years) of financial-analysis experience. All of the participants were either working as financial analysts or had professional financial-analyst experience. Eight of the users were professionally trained to analyze data or to detect fraud. Of the 10 analysts, six were male and four were female.

At the beginning of the study session, participants were asked to fill out a demographic form and were then trained on the use of WireVis for approximately 10 minutes. Participants were also provided with a one-page overview of the functionality of WireVis and encouraged to ask questions. After the training, participants were asked to spend 20 minutes using WireVis to look through the data set for sus-

picious activities. We asked participants to think aloud to reveal their strategies. We specifically encouraged them to describe the steps they were taking, as well as the information used to locate the suspicious activities. Once the users drilled down to a specific transaction, they were asked to record and report their findings on a discovery sheet. Once they documented a specific transaction, they were encouraged to continue looking for others until they reached the time limit. After the exploration process, participants were asked to describe their strategies and additional findings in a post-session interview.

Several methods were used to capture each session as thoroughly as possible. Commercial usability software captured the screen. A separate microphone recorded the user’s audio during the session. Finally, functions built into the WireVis system captured the user’s interaction with the tool as information relevant only to the WireVis system. Instead of recording every mouse movement or keystroke, WireVis captures events that generate visual changes in the system. For example, a mouse movement that highlights a keyword in the heatmap view will generate a time-stamped event.

4.2.1.3.2 Transcribing

Participants’ videos and “think-alouds” were used to create a detailed textual timeline of what participants did during their sessions, along with their self reported reasoning and thinking process. Although the created textual timeline is an interpretation and might not perfectly reflect the participant’s (internal) reasoning process, it was created on the basis of the facts recovered from video and audio with conscious effort to minimize human bias. We therefore consider the resulting transcript to

represent the ground truth of what each participant did during the WireVis analysis.

During the transcribing stage, we identified different findings, strategies, and methods analysts used when investigating fraudulent activities:

- A finding represents a decision that an analyst made after a discovery.
- A strategy is the means the analyst employed to arrive at the finding.
- A method captures the link between finding and strategy. It focuses on the steps the analyst adopted to implement the strategy for discovering the finding.

In a typical investigation, an analyst’s strategy might be to search for a specific suspicious keyword combination on the basis of his or her domain knowledge. For example, the analyst might determine accounts and transactions involving two keywords, “Mexico” and “pharmaceutical,” to be potentially suspicious. Using this strategy, the methods employed by this analyst could consist of a series of actions such as highlighting or filtering those keywords, and drilling down to specific accounts and transactions. At the end of the investigation, the analyst would record his or her findings based on the encountered account numbers and transaction IDs along with the decision about whether the particular finding was suspicious.

Coding We asked several people familiar with WireVis to view each participants’ interactions and determine their reasoning. Specifically, we recruited four coders from our university, all of whom were familiar with WireVis (three male, one female). They used the operation and strategic-analysis tools to view participant interactions and created an outline of what occurred.

We gave all coders comprehensive training on how to use the operation and strategic-analysis tools to examine analysts' interaction logs. We also provided a guideline of hierarchical coding procedures, asking coders to provide hierarchical annotations in free-text format within the VA tools. The hierarchies are reflected as different levels of decision points and strategies extracted by the coders. We asked coders to identify and label findings, strategies, and methods for each analyst. In addition, coders were encouraged to annotate the transitions if they could discover relationships between each decision point, such as one strategy leads to multiple findings or one finding transforms into a new strategy.

All findings from the coders were recorded as annotations and linked to corresponding interaction events and time ranges. Each coder went through the 10 analysts' interaction logs one by one using the VA tools, spending an average of 13.15 minutes reconstructing each analyst's reasoning process. Thus, at the end of the coding phase, we collected 10 sets of annotations from each coder, resulting in 40 sets of annotations overall.

Grading Our next step was to compare the annotations that the coders produced to the ground truth to determine how much of the reasoning process the coders could reconstruct. The comparisons are graded according to a set of predetermined criteria.

The categories we used in grading were in accordance with both transcribing and coding: finding, strategy, and method. Generally speaking, strategy and finding do not necessarily have a one-to-one mapping relationship because some strategies might lead to multiple or null findings. But one finding always comes with a method in the

sense that a method is always needed to make a decision.

For each finding, strategy, and method, we graded according to the following criteria:

- correctly identified - that is, a coder both noticed and correctly identified a meaningful finding, strategy, or method;
- incorrectly identified that is, a coder noticed some meaningful interactions but incorrectly interpreted them;
- false detections - that is, a coder thought that a certain action took place when in fact none had; and
- never identified that is, actions took place but were not noticed or annotated by the coders.

We chose this combination because the four measurements cover all possible scenarios and yet are explicitly distinguishable.

Table 2 illustrates the overall grading criteria. We determined that a finding was correct as long as the coders correctly identified that there was a decision made during the analyst's investigation. But we did not ask them to determine the outcome of that decision (whether the transaction is suspicious, not suspicious, or inconclusive). Additionally, if only a part of the coder's annotation was correct - for example, if the coder determined that a strategy was to look for five incompatible keywords but only identified four keywords correctly we graded that annotation as incorrectly identified. Such strict grading criteria minimizes potential bias in the grading process.

4.2.1.4 Results

The quantitative and observational results obtained from grading are rich and informative. We demonstrate quantitatively the amount of reasoning that can be extracted from analyzing interaction logs and describe some of the coding process's trends and limitations.

4.2.1.4.1 How Much Reasoning Can We Infer?

Figure 30 shows the average accuracy of each coder's reconstructed reasoning processes for all participants. The results indicate that it is possible to infer reasoning from user interaction logs. In fact, on average, 79 percent of the findings made during the original investigation process could be recovered by analyzing the captured user interactions. Similarly, 60 percent of the methods and 60 percent of the strategies could be extracted, with reasonable deviation between the coders.

Across participants A different perspective from which to examine the results is to look for variations in accuracy across the 10 participants. Figure 31 (next page) shows

Table 2: Average time spent in solving each task.

Analyst activity	Ground truth	Grading criteria		Coder		
			1	2	3	4
Finding	6	Correctly identified	5	3	4	5
		Incorrectly identified	0	1	0	1
		False detections	0	0	2	0
		Never identified	1	2	2	0
Strategy	3	Correctly identified	3	3	1	0
		Incorrectly identified	0	0	1	3
		False detections	0	0	1	1
		Never identified	0	0	1	0
Method	6	Correctly identified	6	4	3	3
		Incorrectly identified	0	0	1	2
		False detections	0	0	0	0
		Never identified	0	2	2	1
Time spent (minutes)			14.70	33.39	10.27	35.90

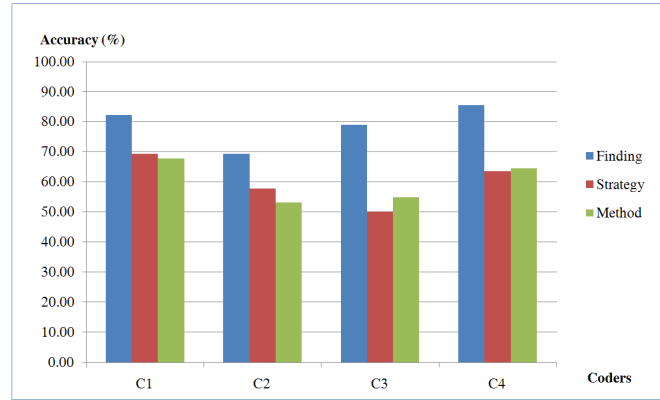


Figure 30: The average accuracy of the four coders in correctly identifying findings, strategies, and methods of the 10 participants. Analyzing the captured user interactions recovers an average of 79 percent of the findings, 60 percent of the methods, and 60 percent of the strategies, with reasonable deviation between coders.

the average accuracy of the coders in recovering the 10 participants' reasoning processes. This result indicates a noticeable difference between accuracies in extracting reasoning processes for different participants. This finding leads us to conclude that some analysis processes are more difficult to follow than others. Although there is no definitive explanation for this difference, our investigation suggests two plausible contributors.

The first factor is the difference in experience with financial fraud detection between the participants and coders. Because our coders have no fraud detection training, it is natural that some of the strategies and methods in investigative processes are lost to them.

The second reason for this variation is manifested in the acute drop in accuracy when extracting methods from the analyses of participants 2 and 4, as shown in Figure 31. As the figure suggests, the coders were baffled by the methods of these two participants. Upon investigating the video of these participants' analysis process,

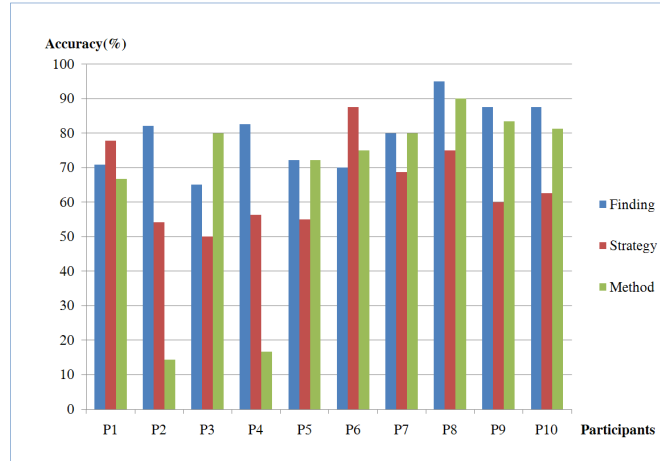


Figure 31: The average accuracy of correctly identifying findings, strategies, and methods based on the 10 participants. As the graph shows, accuracy levels in extracting reasoning process for different participants varied significantly. We hypothesize that this variance is due both to differences in experience between participants and coders and to the difficulty of following some analysis processes.

we discovered that participants 2 and 4 focused on the irregularities in the WireVis time series view. Specifically, they closely examined spikes in the view (see Figure 32), which indicate sudden increases in amounts or frequencies of wire transactions. Our coders had no way of seeing these visual patterns, so they could not identify the methods behind the participants' analyses.

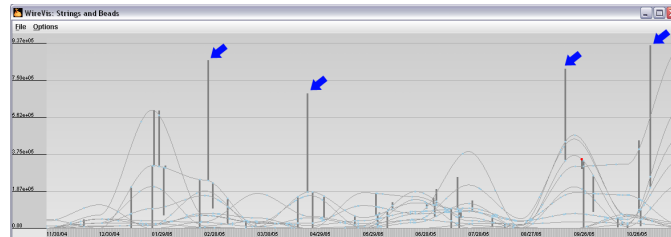


Figure 32: The time series view in WireVis showing spikes that indicate sudden increases in the amounts or frequencies of wire transactions. Because the coders could not see these visual patterns, they could not determine the methods behind the participants' analysis processes.

Considering false detections Because this study aims to determine how much of the reasoning process can be extracted from interaction logs, we have reported the accuracy based purely on the number of correctly identified elements. However, it is relevant to note how often our coders' detections turned out to be inaccurate. Under our grading scheme, the number of annotations made by a coder often exceeded the number of elements in the transcription because of the false detections. For example, the grading result for Participant 1 in Table 2 shows that the number of findings in the ground truth is 6; however, Coder 3 made a total of eight annotations. This coder correctly identified four of the six elements, missed identifying two of the six elements, and made two false detections.

With the false detections in mind, we reexamine the coders' accuracy not on the basis of how much of the reasoning process can be recovered, but on the basis of the accuracy of their annotations. Figure 33 shows the result of the coders' accuracies that include the coders' false detections. Not surprisingly, the accuracy of all the coders decreases slightly. Their accuracy in extracting findings drops by 3 percent, from 79 to 76 percent; strategies by 5 percent, from 60 to 55 percent; and finally methods, by 2 percent from 60 to 58 percent.

4.2.1.4.2 Amount of Time Spent by Coders

One important aspect in extracting reasoning processes is the amount of time needed to analyze the interaction logs.

Capturing time spent by a coder Built into our operation and strategy analysis tools is the ability to track the amount of time that a coder spends using the tools. The

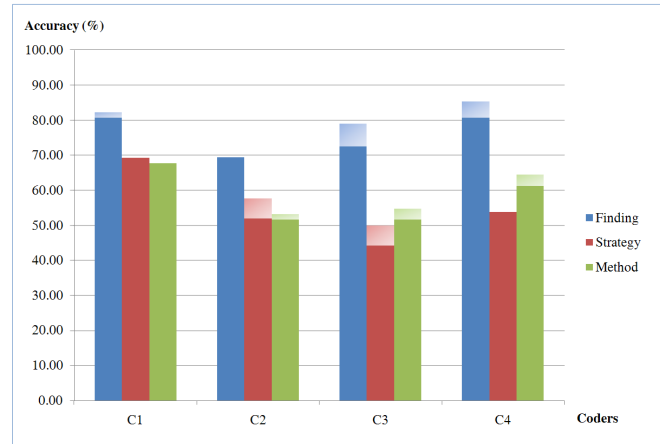


Figure 33: The accuracy of the coder’s annotations in matching the analysts’ findings, strategies, and methods. The semitransparent areas indicate the decrease in accuracy compared to Figure 30. Whereas Figure 30 indicates the amount of reasoning that can be recovered, this figure shows the accuracy of the coders’ annotations.

coders were made aware of this feature and were told not to take breaks during an analysis. Because the coders directly annotated their discoveries into the operational-analysis tool, the system was able to record the amount of time spent by each coder when analyzing an interaction log.

Furthermore, the system tracked when the coder started and stopped the annotations. The purpose of this feature was to separate the time spent analyzing the interaction log from the time spent annotating. On average, the coders spent 23.9 minutes analyzing one interaction log. Of this amount, 10.75 minutes were spent on annotation and the remaining 13.15 minutes on investigation.

Time spent versus accuracy Overall, there is no correlation between the time spent by a coder and accuracy. Figure 34 plots the relationship between the coders’ time spent in analysis (not including time spent for annotation) and their accuracy in extracting findings. With the exception of the two outliers on the far right, it appears

that the coders are consistently successful when spending anywhere from five to 15 minutes. This suggests that spending more time in the analysis does not always yield better results. The two outliers represent the analysis of Coders 2 and 4 in their first investigation (Participant 1). All coders became more proficient in their analysis as they gained experience.

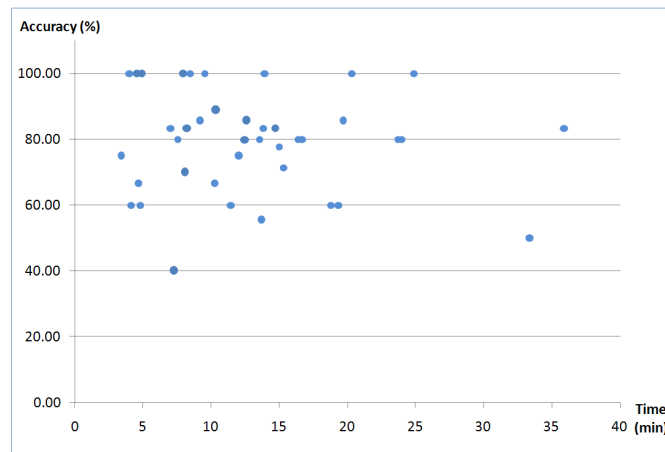


Figure 34: The accuracy of the coders in recovering findings of the participants and the amount of time spent. The results show that coders generally can accurately extract findings when they spend between five and 15 minutes on the analysis.

Increase in accuracy. As Figure 31 shows, the coders' accuracy increases as they gain experience investigating interaction logs. All four coders began with examining Participant 1's interactions and ended with Participant 10's. Using Pearson's correlation coefficient, we find that the number of participants a coder has examined is positively correlated to the coder's accuracy. This correlation is statistically significant when extracting findings ($r(40) = 0.37, p < 0.05$) and methods ($r(40) = 0.52, p < 0.01$). Only in extracting strategies is the correlation weaker ($r(40) = 0.21, p = 0.182$). Although the sample size is relatively small, these statistics imply a subtle but potentially important discovery: with more experience analyz-

ing interaction logs, a coder could become more proficient in extracting an analyst's reasoning process.

4.2.1.5 Discussion

The study we've described is complex and intricate. Although many of the nuances encountered during the study do not affect the results and therefore are not described here, some findings might be of interest to the community. First, during our informal debriefing, the coders discussed the strategies that they used in analyzing participants' interaction logs. The coders often began their investigations by looking for gaps in the operational-view timeline (Figure 28), which are the by-products of the analysts taking time to write their findings on the discovery sheet. The coders used these gaps to look for the analysts' findings and then worked backward to discover the strategies and methods used to derive the findings.

Although this strategy might seem specific to this study and non generalizable, in a real-life scenario, analysts either directly make annotations in the visualization to note a finding, or write their finding on a piece of paper for future reference. Either way, there will exist a visible marker that suggests a relevant discovery by the analyst. Therefore, although we did not anticipate this strategy by the coders, we found their quick adoption of this method to identify the analysts' findings to be effective and relevant.

A second interesting trend pointed out by our coders concerns the usefulness of our visual tools for depicting the analysis's operational and strategic aspects. All of the coders used the operational analysis tool first to get an overall impression of an

analyst's interactions. However, the strategic analysis tool is often used to examine a specific sequence of interactions when the interactions appear random and jumbled. By presenting the results of the interactions from three perspectives (accounts, keywords, and time) in the strategic tool, the coder could often identify the focus and intent behind the series of interactions. This finding not only validates our design of the tools but also reconfirms the importance of visualizing both the strategic and operational aspects of an analysis process. In fact, most of the coders began their investigation by looking for gaps in the interactions to identify findings. Next, they looked for strategies by examining the overall visual patterns in the strategic- and operational-analysis tools without focusing on individual user interactions. Finally, methods were extracted using the operational analysis tool, in which specific interactions were examined in detail.

One last relevant aspect of our study is the measurement of incorrectly identified elements in the grading process. None of our results account for elements that have been graded as incorrectly identified. As we mentioned earlier, any annotation by a coder that does not perfectly match the transcription is considered to be incorrectly identified. This includes scenarios in which a coder identifies the analyst's strategy to be examining four keywords when in fact the analyst was examining five, or when a coder determines that the finding of the analyst is a transaction between Accounts A and B instead of Accounts A and C. If we gave half a point to these incorrectly identified elements, the overall accuracy of extracting strategies would increase drastically from 60 to 71 percent, methods from 60 to 73 percent, and findings from 79 to 82 percent.

We found that when analysts based their investigations purely on visual patterns, our coders had a difficult time determining their methods. Although some of the coders' errors in extracting the analysts' reasoning process can be attributed to operator errors, the most consistent and common errors stem from the coders not being able to see the same visual representations as the analysts. This observation reveals a potential pitfall of examining interaction logs without considering the visual representations. Our current approach is thus probably limited to highly interactive visualization systems. Understanding how interactivity versus visual representation affects reasoning extraction remains an open question.

One practical solution is to connect the operational-analysis tool directly to the video of the analysis. With the operational-analysis tool functioning as an overview, coders can review videos of segments of interactions that are ambiguous to them. If analysts used the operational-analysis tool to aid the recall of their own analysis process, the video could further serve as a record of the details of the original investigation.

By combining video with the operational-analysis tool, we believe that coders can achieve a higher degree of accuracy and in turn be able to derive winning strategies of different analysts that lead to the same findings. By combining these winning strategies, we hope to identify critical decision points that these strategies share and uncover the necessary reasoning process for identifying a particular type of fraudulent activity.

Finally, we plan to analyze the difference between groups of participants with diverse backgrounds. Our previous study involved participants who are not trained in

financial fraud detection [33]. Although they were also able to point out suspicious events and activities, we wish to compare their decisions with the findings of real financial analysts. If we can discover some common pitfalls in novice analysts’ reasoning processes, we believe we can create better training tools to help these novices become proficient more quickly.

4.2.2 Voting Viz: Perception of agreement or disagreement of two or more ideas

In recent years, several researchers have considered adopting knowledge in their visualizations. Among them, Chen et al. [27] proposed a knowledge-assisted visualization model based on the differentiation of data, information, and knowledge in visualization, in which both existing and users’ personalized knowledge are used to design a visualization application. While their model has been shown to be novel and effective, it is not clear how such the model could be broadened to visualization applications. In this section, we extend the spirit of incorporating knowledge into visualizations and propose a new approach to incorporate users’ knowledge into visualization. In here, we use a restricted definition that knowledge is the “perception of agreement or disagreement of two ideas [21].” In this fashion, knowledge is regarded as the perception of the connection of agreement, or disagreement and repugnancy of any ideas.

Previously, we performed an expert evaluation to see how user interactions encode experts’ reasoning (see section 4.2.1 for detail). From the study, we observed that financial analysts have different opinions on each financial transaction. Although they have different ideas and sometimes contradict each other on determining each trans-

action as suspicious, unsuspicious, and inconclusive, all their ideas are important. To preserve both the analysts' interactive analysis and their understanding on financial transactions, we designed a system called Voting Viz. In here, we consider representing both consistent and contradictory knowledge in our visualization application.

4.2.2.1 Interface Design

The overall interface of our system (Voting Viz) is designed on top of the WireVis system [27]. The updated version of WireVis (called WireVis-CE, Figure 35 (left)) is used. Comparing to the original WireVis system, WireVis-CE consists of several touchable buttons to be operated on a multi-touch table. Although the WireVis system has been known to be a well-designed visual analytics system, it has the limitation of only showing direct relationships between accounts, keywords, and transactions because the system focuses on representing the hierarchies of clusters. If the user wants to see the relationship between a specific transaction and accounts or keywords, she needs to browse the hierarchy from the highest-level down to the level of individual transactions for the desired cluster of interest. A Relation Window (Figure 35 (right)) is designed differently, with which the user can directly observe the relationship by simply performing a dragging operation on accounts, account, or keywords.

4.2.2.1.1 WireVis-CE

Similar to WireVis (Figure 27), WireVis-CE (Figure 35 (left)) consists of three views including a Heatmap view (Figure 35 (F)), a Strings and Beads view (Figure 35 (G)), and a Keyword Relation View (Figure 35 (H)). All functionalities in WireVis are embedded into WireVis-CE. The difference between the two is that instead of having

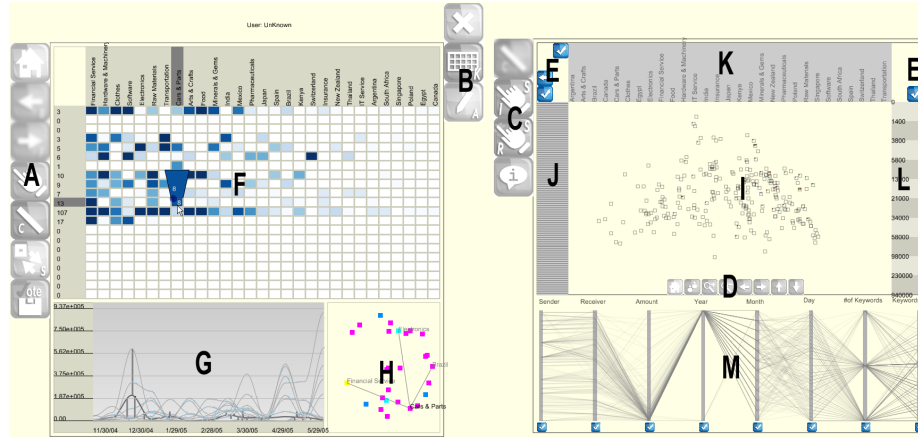


Figure 35: Overview of Voting Viz. It consists of an updated version of WireVis (called WireVis-CE (left)) and a Relation Window (right). WireVis-CE is designed similar to WireVis system except of having buttons (A and B). And the Relation Window consists of multiple coordinated views (I ~ M) to show the relationships between keywords, accounts, and transactions. The Relation Window also has buttons (C ~ D) to control the interface.

a traditional menu system, WireVis-CE has touchable buttons to make it work on both a desktop computer and a multi-touch table.

The Heatmap View and the Strings and Beads View can be switched such that the Heatmap View takes up the lower left hand position and the Strings and Beads View fills the main display. This simple switch operation allows the user to utilize the visual real estate for focusing either on a hierarchical projection of data or to examine in detail all transactions in the Strings and Beads View. In Voting Viz, each vote is identified by the user's name displayed above WireVis-CE. The user's name also can be changed by simply clicking the displayed name.

4.2.2.1.2 Relation Window

The design of the Relation Window is based on multiple coordinated views: the Projection View (Figure 35 (I)), the Information View (Figure 35 (J ~ L)), and the

Data View (Figure 35 (M)). The Projection View represents a specific aspect of the financial transaction data that are coordinated in such a way that any interaction with one view is immediately reflected in all the other views (brushing & linking).

Projection View The Projection View projects all transaction data points onto a two-dimensional coordinate system. By default, first two principal components (the first and second most dominant eigenvectors) are used. To find the principal components, Principle Component Analysis (PCA) is used. By default, all 8 dimensions are used to find the principal components. Selectively, unwanted dimensions can be excluded from the calculation of principal component analysis by toggling buttons located at the bottom in the Data View. In this view, a zooming function is applied to increase the readability of data.

Information View The Information View has three information panels as accounts ((Figure 35 (J))), keywords (Figure 35 (K)), and the amounts of the transactions (Figure 35 (L)). Relevant information of highlighted or selected transactions are visualized in these panels as connected lines. Showing the relationship can be selectively enabled or disabled by pressing the check buttons located beside or above of each information panel (Figure 35 (E)).

Data View The Data View is located below the Projection View, and shows a parallel coordinates visualization of all transactions in the original data dimensions. There are about 8 dimensions in financial transaction data as sender account, receiver account, the amounts of the transactions, date information (year, month, and day), number of keywords, and actual keywords related to each transaction. To perform

the principal component analysis, all keywords information (i.e. categorical information) in each transaction are converted to a decimal format. To make the decimal format, a simple “binning” technique is used. For instance, n bit patterns are generated depending on the total number of keyword (n). Each bit pattern represents an individual keyword bin. All bit patterns are initialized with a binary number “0.” Based on the keyword information, the corresponding bin is mapped with a binary number “1.” Then, the mapped bit patterns are converted to a decimal number to represent unique keyword identification number.











4.2.2.1.3 Menu buttons

As shown in Figure 35, a total of 10 menu buttons for WireVis-CE and 14 menu buttons for the Relation Window are designed to support analysts’ interactive analysis.

In WireVis-CE, six buttons are for interacting with represented data items (Figure 35 (A)) and three buttons (Figure 35 (B)) for performing annotation operations. And in the Relation Window, first three buttons in Figure 35 (C) are for interacting with represented data elements, the last button in Figure 35 (C) is for showing account information for voted transactions, eight small buttons in Figure 35 (D) are for controlling the Projection View, and toggle buttons in information panels (Figure 35 (D)) are for showing detailed information about highlighted or selected items in the Projection View.

Table 3 represents the menu buttons and their meanings. With each menu button, the user can initiate an operation of selection, navigation, etc.

(a)

Button	Meaning	Button	Meaning
	Go back to the initial state		Switch between Heatmap View and String and Bead View
	History backward		Initiate voting
	History forward		Close the workspace
	Navigation on hierarchical clusters		Text annotation
	Cancel the selected button(s)		Drawing annotation

(b)















Button	Meaning	Button	Meaning
	Cancel the selected button(s)		Show the clusters of voters
	Individual item selection		Zooming in
	Range item(s) selection		Zooming out
	Show account information on voted transactions		Move left
	Toggle selection of enabling or disabling		Move right
	Show transactional information between sender and receiver account		Move up
	Go back to the initial state		Move down

Table 3: Menu buttons used in WireVis-CE (a) and the Relation Window (b).

4.2.2.2 Voting and Preserving the Users' Reasoning

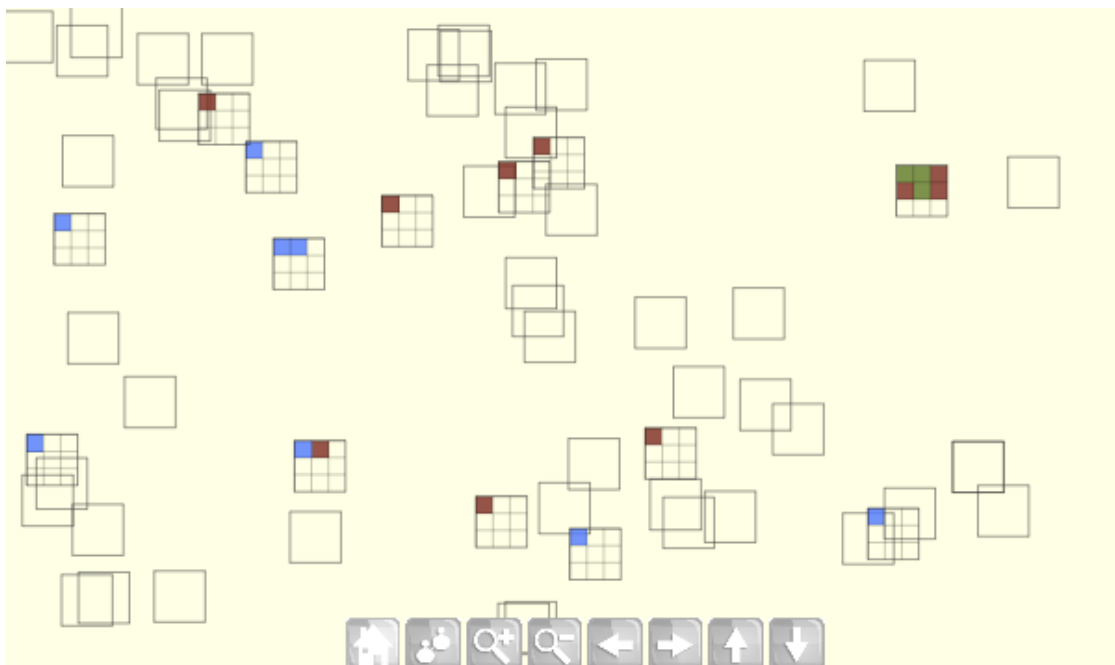
In visual analytics, managing the users' findings is as important as providing useful analysis tools because their end goal is to find evidence that verifies or supports their hypothesis. By investigating wire transactions, financial analysts often determine the investigated transaction(s) as suspicious, unsuspicious, or inconclusive. Since their decisions are often based on experiences, it is difficult to understand how they ended up with their decisions. Voting Viz is designed to show their decisions as well as their reasoning. The reasoning is managed as a screenshot or a detailed description (annotation). The screenshot shows what has been found from the analysis and the detailed description explains the reasoning.

Voting in Visualization As mentioned previously, Voting Viz allows the user to explore the financial transaction data in WireVis-CE and examine the relationships further in the Relation Window. From the analysis, it is important to preserve all analysts' decisions (fraudulent or not fraudulent) in visualization.

In the Relation Window, all expert analysts' decisions (or findings) from the previous study (see Section 4.2.1 for detail) are embedded to the Relation Window. The analysts' diverse opinions on each transaction are displayed inside the Projection View and colored as red ("suspicious"), blue ("unsuspicious"), and green ("inconclusive"). As shown in Figure 36, a grid mapping method ($n \times n$) is used. The value n is determined by finding the smaller integral value that is not less than the square root of the maximum number of findings (x) as:

Transactions Discovery Sheet						
Suspicious NOT Suspicious Inconclusive	Sender Account Number	Receiver Account Number	Transaction Keywords (abbreviated)	Transaction Date	Transaction Amount	Count
S	80	44	Food For Services	10/30	284,40.00	1
N	154	69	Raw.	07/21	109,000	2
S	154	64	Cash + Parts	02/01	600,000	3
N	142	100	Finan Sen	06/24 02/01	525,000	4
S	8	172	Finan. Serv.	8/22	605,263	5
N	141	144	Clothes, Finan Sen.	8/22	341,082.03	6
S	8	172	Fin Services	8/22	605,263	7
S	154	128	Car Parts	2/07	600,000	8
N	82	5	Fin. Sen.	7/12	20,000	9
N	1	142	Software	11/28	9070.00	10
						11

(a)



(b)

Figure 36: During the experts evaluation, financial analysts were requested to write down their final findings in transaction discovery sheet. Figure (a) shows an expert's transaction discovery sheet. Experts' findings in transaction discovery sheets are displayed within each transaction in the Projection View (b). A simple color mapping method is used to highlight suspicious (red), unsuspicious (blue), and inconclusive (green) findings.

$$[n] = \max\{\varphi \in \mathbb{Z} | \varphi \leq n\} \quad \text{for all } \varphi = \sqrt{x} \quad (9)$$

where x and φ are real numbers and n is integer, and \mathbb{Z} is the set of integers.

In Voting Viz, users can interactively vote on transaction(s) while performing interactive exploration in WireVis-CE. To perform the voting in visualization, the user can simply toggle the voting button (located at the bottom in Figure 35 (A)) and determine the investigated transactions as suspicious, unsuspicious, or inconclusive. With each voting action, the Projection View updates the voting information correspondingly.

Managing the Reasoning Instead of simply voting on transaction data, it is important to track analysts' reasoning. A simple approach has been designed to capture a screenshot with additional explanations. To capture screenshots within the collaborative workspace, OpenGL frame buffer is saved. Two annotation methods are designed and supported within the collaborative environment, text- and drawing-annotation. Figure 37 shows two examples, in where the user describes what and how they found directly on screen. For the text-annotation, a keyboard or a virtual keyboard can be used, and a mouse (i.e. pointing device) is used for the drawing-annotation.

In Voting Viz, all users' reasonings are managed in an XML format similar to the P-Set model [70]. The parameter sets we used are similar to the sets defined in the P-Set model. However, we define each parameter to work for our system. When the voting action is performed, the current screen is captured and managed as an individual vote. Since the user's reasoning can easily be recreated with the parameter sets, each user

can easily grasp other analysts’ reasonings from the Relation Window and recreate to see in WireVis-CE.

Voting Viz is designed with high interactivity in mind. In both WireVis-CE and the Relation Window, several types of interactions are designed. All these interactions include selection, zooming, panning, etc. Although such interactions are standard in most InfoVis or visual analytics tools, they are nonetheless very important, and are essential in multiple coordinated views.

To support high or low-resolution display, the scale of WireVis-CE and the Relation Window can be expanded or shrunk. With the supported interactions in Voting Viz, the user can quickly move the mouse over the data to see detailed information (i.e. mouse hovering). In WireVis-CE, for example, when the user moves the mouse over the keyword names or account clusters in the heatmap, corresponding cells in the

heatmap are highlighted along with the number of occurrences of the transactions in each account cluster. At the same time, all the beads in Strings and Beads are highlighted. Similarly, in the Relation Window, the user can move the mouse over the data in the information panels and projected data elements in the Projection View. This allows the user to focus on specific transactions and observe which accounts are transacting over what keywords and amounts. Such tightly integrated interactions are supported in the Information View, the Projection View, and the Data View.

All interactions in WireVis-CE allows the user to perform rapid exploration over many accounts, keywords, and time ranges. In addition, in the Relation Window, the interactions permit rapid examination through seeing relationships between accounts, keywords, and the amounts of the transactions. The ability to select data items in one view and immediately see the corresponding items highlighted in the other coordinated views helps the user to understand the relationship between selected items in more than two views.

The most notable interactions in this category are the different types of selections implemented in the Relation Window. The user is allowed to select data items in all three views. In the Projection View, selection means clicking on a single dot (individual elements) or draw an enclosed space upon which all data items within the space will be selected. In the Data View, the user can either click on a single line or brushing a range of items. In the Information View, the user can only click on a single item. Since account information (i.e. account number) are not displayed because of the limited display space, selection in the account information panel is designed to be performed with two mouse actions; first the user holds down the mouse left button

and moves over the account information panel to see the detailed account information, and then releases the mouse button to select.

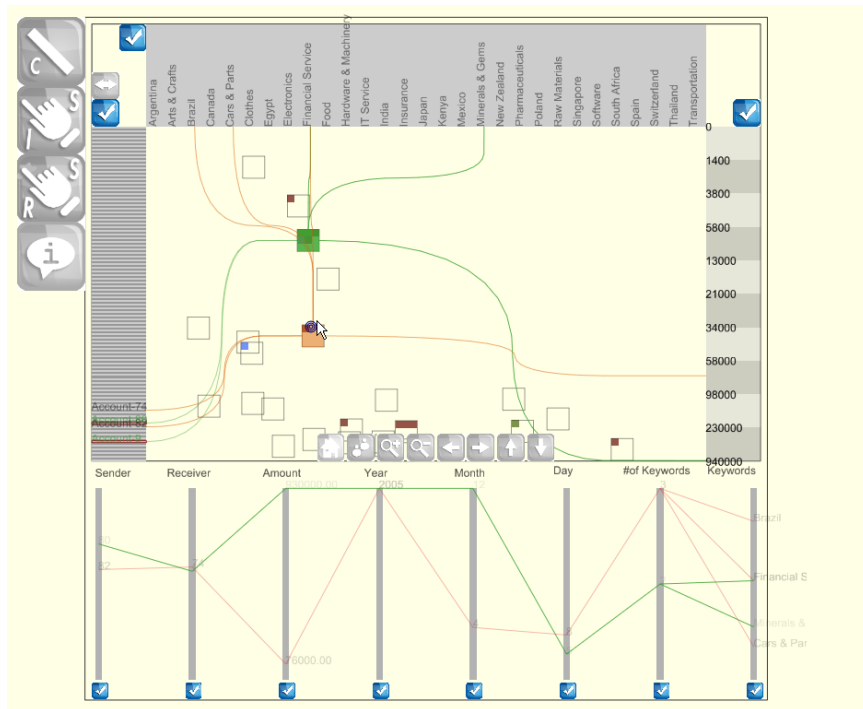


Figure 38: Simple selection and dragging operations in a zoomed-in space. With selecting an item (highlighted in green), the user can perform further analysis to find similar transactional patterns (highlighted in orange).

Figure 38 shows an example of performing three operations in the Relation Window. After performing a zooming operation to see voting information on a specific wire transaction in detail, an item is selected by a single item selection operation to find other transactions having similar patterns.

As shown in Figure 39, there are several additional features of interactive examinations in the Relation Window. Figure 39 (a and b) show that the user examines the transaction (from account 80 to account 9). From this view, the user sees the overall transactions related to account 80 and account 9. After performing the zooming-in operation, the user can observe that five voters had contradictory opinions on this

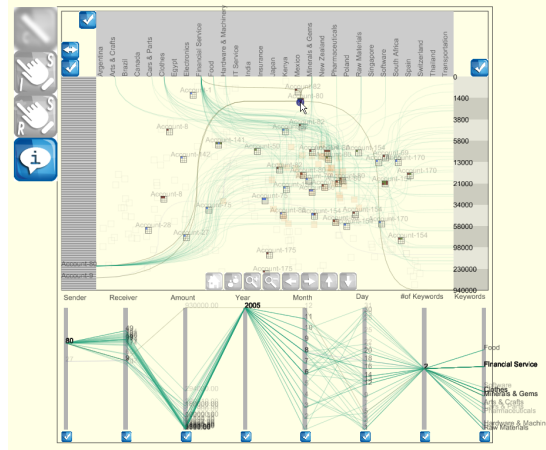
specific transaction as two suspicious, one unsuspicious, and two inconclusive. From the Figure 39 (b), it can also be observed that this particular transaction is of an amount much larger than others.

In addition, the user can interactively examine transactions in the Data View. The transparency of keyword names in this view is set based on the frequency of keywords that occur in that transactions of the accounts (Figure 39 (a)). With this feature, the use can verify that account 80 made a lot of wire transactions relating to the keyword “Financial Service.” For this account, although the overall frequency of “Clothes” and “Minerals & Gems” is a bit less than the keyword “Financial Service”, it can be observed that two keywords “Clothes” and “Minerals & Gems” occur on numerous occasions.

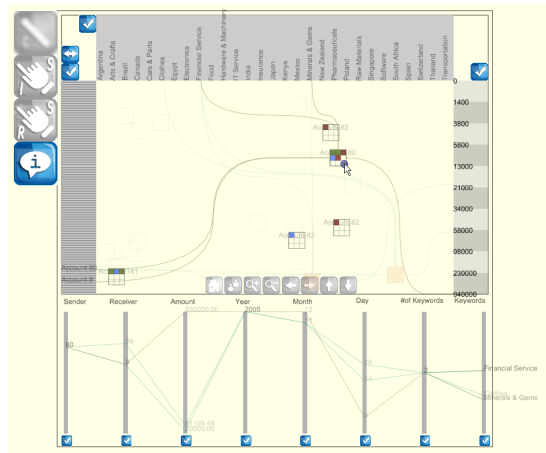
Since the purpose of designing Voting Viz is to help the users to understand the voters’ analysis patterns on wire transactions, showing each voter’s analysis pattern as a cluster is useful to understand voters’ behaviors. Figure 39 (c) shows that all 10 experts’ analysis patterns in the Projection View. From this figure, it can be observed that each of their analysis patterns is concentrated in a certain region in the PCA space. This is because most their analysis patterns began by exploring the number of keywords and the amounts of the transactions in Heatmap or Strings and Beads Views.

4.2.2.4 Discussion

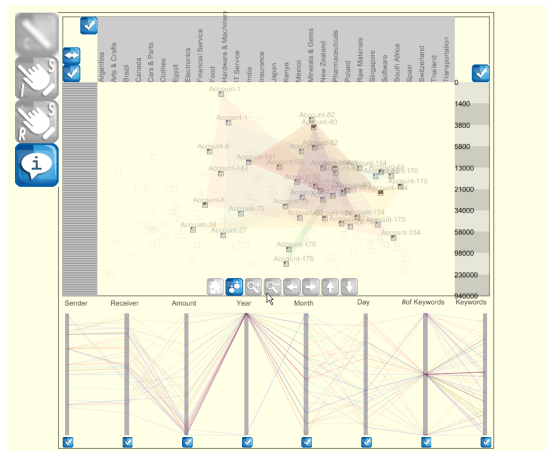
In Voting Viz, financial analysts can perform exploration and examination of wire transactions to make decisions on transactions as suspicious, unsuspicious, and incon-



(a)



(b)



(c)

Figure 39: In the Relation Window, interactive examinations are performed. Both (a) and (b) show that the user highlights the same transaction data (account 80 sending money to account 9) in different scales. (c) By toggling a button (toggled button is highlighted in blue) in the Projection View, clusters are created using the voters' names to highlight the voters' analysis patterns.

clusive. The provided set of interaction techniques are useful to perform interactive exploration as well as examination on wire transactions. Although their decisions are diverse and sometimes contradictory to each other, all their decisions are important. Instead of keeping only important decisions on transactions, Voting Viz focuses on preserving all users' decisions. In addition, their reasoning can be traced with annotation techniques.

We believe that Voting Viz is well designed to address the analysts' overall analytical processes. However, it is necessary to conduct an evaluation to validate the system as well as our idea of preserving all users' findings in visualization. In Voting Viz, we adopt a simple approach (annotation) to capture the analysts' reasoning. However, how much the analysts' reasoning can be recovered from the annotation is unclear. Further study should be performed to understand the relationship between annotation and analysts' reasoning.

4.3 Combination

Combination is the process of systemizing explicit concepts into an explicit knowledge system [97]. Since explicit knowledge exists everywhere in books, research papers, and communication networks (user groups), etc., the process of combining different bodies of explicit knowledge is important. For instance, genomic data have been used in many different research areas: biology, bioinformatics, computer science (including visualization), health & medical science, etc., and depending on the domain, researchers have derived different, yet equally important findings. In order to fully comprehend the knowledge associated with such genomic data, it is neces-

sary to combine findings from different domains and integrate them into a cohesive set of explicit knowledge. In visualization, there is no known system that supports the combination process. However, there exist some institutions such as NCBI (The National Center for Biotechnology Information) that provide and support the combination process. Researchers can post their findings through the web site with their journal publications. Based on the collection of the research works (findings), NCBI summarizes (combines) and posts the findings as explicit knowledge on the web.

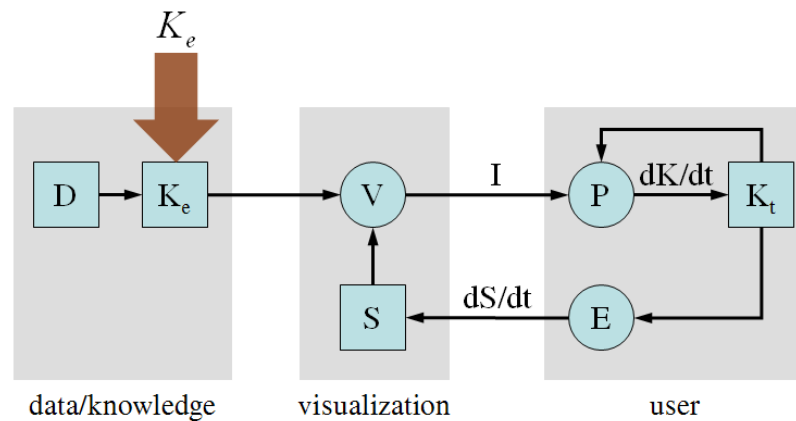


Figure 40: Combination process (indicated by the red arrow). Explicit knowledge can be combined with existing explicit knowledge in a knowledge base to further enhance a user's visual analysis process.

Figure 40 shows a simple model of combining explicit knowledge into an existing knowledge-base. Behind this simple diagram, however, additional considerations need to be addressed in order to maintain the quality and integrity of the knowledge-base when combining new explicit knowledge with an existing source. If unrelated or incorrect knowledge is combined with the existing explicit knowledge, it could degrade the overall trustworthiness of the knowledge-base as well as the benefits of representing knowledge in a visualization. While we are not aware of any known visualizations

that support the verification and validation process when combining explicit knowledge, in the Knowledge Engineering literature, researchers have long been studying how to verify and validate underlying knowledge when developing a knowledge-based system [140]. The specifics of knowledge management and engineering is outside the scope of this section, but it should be considered.

If new knowledge is not carefully validated, inserting unrelated or incorrect knowledge could potentially degrade the value of the ontological knowledge structure. However, verifying and validating diverse human knowledge is difficult in nature. Due to individual experiences and understanding, different experts have their own ways of representing and describing knowledge. In general, there are three potential issues in inserting newly created knowledge into an existing knowledge database: duplicated, partial-overlapped and conflicted knowledge.

- Duplicated knowledge is often created when different experts share the same discoveries without realizing that one already exists. In such case, an expert user should carefully address the issue by merging the two into one or creating an identical one (but having a slightly different meaning).
- Partially overlapped knowledge in the knowledge database represents rules that are mostly similar but contains subtle yet important differences. Since it is not simple to decide if the newly added rules should be inserted or merged with the existing one, verifying this type of relationship is difficult. One conservative approach could be to provide multiple copies of the overlapped rules and group them together for individual users to choose from.

- Conflicted knowledge occurs when existing knowledge differs from newly created knowledge. This happens when the knowledge was true in the past, but not any more in the present time. Organizing the conflicted knowledge is challenging, but it should be carefully resolved by collaborating with domain experts.

Understanding each of these relationships is imperative for maintaining the validity and integrity of a knowledge-base that is used in real decision-making environments. At present time, all three scenarios are handled by domain experts manually. However, in Knowledge Engineering literature, researchers proposed and designed several verification and validation (V&V) techniques and tools [140]. Some of them support the ability to automatically verifying and validating underlying knowledge. But without a clear understanding of domain knowledge, most automatic techniques and tools are not always reliable. In knowledge visualization, it remains an open research area for us to create a semi-automated knowledge management system for organizing and combining diverse knowledge.

In visualization, adopting existing knowledge is regarded as a common, but important approach to strengthening the visualization. In this section, we provide several examples to show how knowledge can be combined together in visualization.

4.3.1 Adopting Knowledge in Visualization

Most visualization applications are well designed by visualization experts, but understanding the data and the domain knowledge necessary for the appropriate filtering and extracting algorithms are often beyond the expertise of the visualization designer. Collaboration with domain experts alleviates some of these shortcomings, but often

the domain experts' knowledge cannot easily be transferred and directly integrated into the visualizations (sometimes referred to as the “communication gap” [109]). Therefore, it is important to combine existing knowledge into visualization instead of only supporting either addition visualization techniques or redesigning visualization representations. With this approach, the resulting visualization will be enhanced not just visually, but also in actual use for solving domain specific tasks. In here, we provide two examples how we can combine knowledge and apply them to visualization.

4.3.1.1 Examples

Some researchers consider applying knowledge into visualization applications to enhance their visual representations. Xiao et al. [155], for instance, present a network traffic analysis system that reuses users' knowledge (discoveries - the meanings of each network traffic pattern) to change the visual representation in a meaningful way (Figure 41). With limited knowledge about each network traffic pattern, the visually represented network traffic patterns do not have enough information explaining what each pattern means. If the application is modified by adopting users' knowledge, detecting and understanding the meanings of each network traffic pattern become a trivial but meaningful task.

Also our genomic visualization application (called GVis [65]) shows how the existing visualization application can be redesigned to incorporate domain knowledge and enhance the understanding of complex genomic data and finding useful information (Figure 42). The initial version of the application has been designed to represent genomic data and to address biological analysis processes. It has been extended with

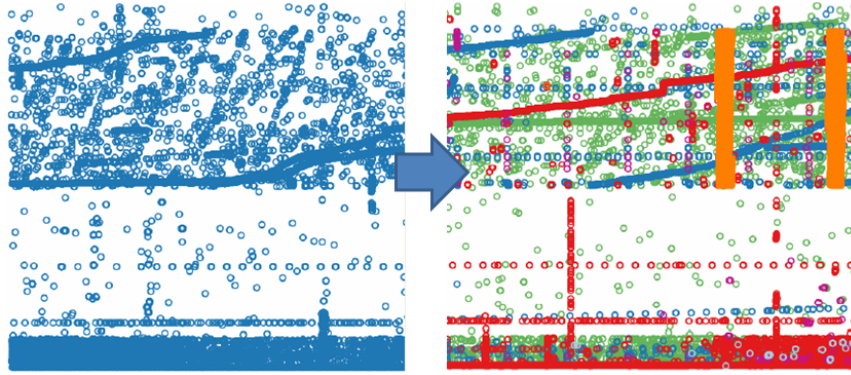


Figure 41: A network traffic analysis system [155] is enhanced by adopting users' discoveries. In here, a color representation technique is used to represent each network traffic pattern such as mail (green), DNS (blue), scan (red), etc.

known biological knowledge structures such as general biological classification and biological taxonomy structure from NCBI to enhance understanding of the biological categorization.

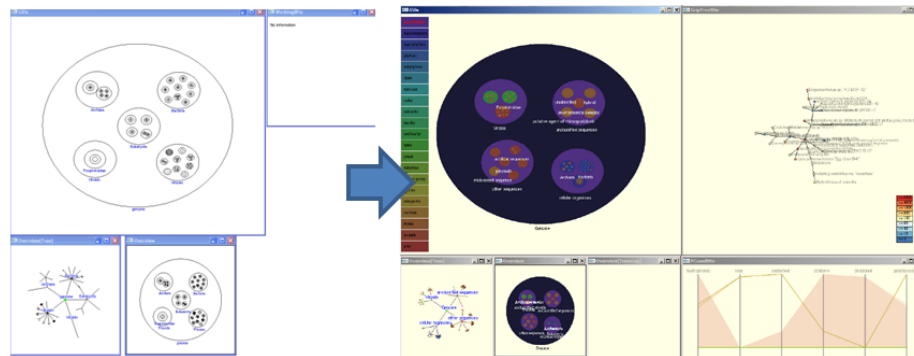


Figure 42: A genomic visualization application (GVis) [65] is extended based on adopting existing biological classification that mapped with each individual color.

The biological classification is a form of scientific taxonomy [90]. The classification is created by biologists based on biological types, such as genus or species. In general, the biological classification forms a hierarchical structure. As shown in Figure 42, the color panel positioned in side of the window represents the biological classification. About 21 color regions are created in the panel. Each color region indicates biological

taxonomic rank such as “Domain,” “Kingdom,” etc. This predefined color maps are used to represent all genomic data. Instead of simply displaying the biological classification, the user can interactively select a region in the color panel to highlight where the relevant genomic data are located in a hierarchical genomic space.

To create the hierarchical genomic space, the NCBI taxonomy structure [149] is used. The NCBI taxonomy structure is designed to organize all GenBank data by referencing existing literatures. The taxonomy structures corresponding to each dataset are fetched by creating a module to communicate with the NCBI server. With the fetched information, all genomic data are organized to form a node-link tree structure. Each node in the structure is directly mapped with a circle in the genomic space (A detailed description about the system is included in section 5.2.1.1). Within the genomic space, the user can interactively navigate through to see the detail about the genomic structures as well as the detailed explanations.

Adopting knowledge into visualizations is important and useful for finding important information. However, identifying relevant and appropriate knowledge that could be applied to a visualization application is difficult. One possible source of knowledge could be extracted from the experts’ analysis. As we have shown in section 4.2.1, the analysts’ analytical processes, strategies, methods, and findings, can be recovered through only the examination of analysts’ interaction log [148]. In the study, 10 analysts’ interactions of analyzing financial fraud were captured in a financial visual analytics application (called WireVis [25]) and analyzed by four coders. The study showed that up to 80% of the findings could be recovered from the interaction logs through the use of specifically designed visual analytical tools [71]. This finding

suggests that the users' interaction logs (e.g. semantic-level interactions such as keywords, accounts, and transactions) partially encode the users' reasoning processes. Therefore, the users' reasoning processes (domain knowledge) can be used to enhance understanding and analyzing data through a visualization application that considers and incorporates the users' interactions.

4.3.1.2 Discussion

As a part of the knowledge combination process, knowledge can be combined and visualized. Visualization applications such as the network traffic analysis system and the genomic visualization integrate knowledge into the existing visualization applications. In this case, the overall visual representation can be enhanced which in turn can increase the users' ability of understanding the data.

Visualization experts realize that combining knowledge and using them in visualizations is important, however the exact process is still unclear in terms of how to find valuable knowledge and how to combine them. In the field of knowledge engineering, researchers have been thinking about how to combine and how to validate knowledge for many years. Although their final goal is different from knowledge visualization, their solutions might be useful for the visualization community to combine, validate, and apply knowledge in creating visual analytical systems.

4.4 Collaboration

Collaboration is the process of sharing tacit knowledge between people. In the knowledge management literature, it is defined as *socialization* [97]. Although both collaboration and socialization represent learning from others, we use the term *collab-*

and visualizations and findings such as using tools like TeraGrid [12], Many Eyes [66], etc (see [84] for detail).

Figure 43 shows how two visualization users could collaborate and share their tacit knowledge with each other. In this diagram, we show that the collaboration process can occur through the use of collaborated visual environments. However, the most natural method for sharing knowledge is still direct communication between the users (via phone, email, instant messages, etc.). In either case, the users are actively sharing their discoveries and tacit knowledge and incorporating each other's domain expertise into their own.

How collaborations take place within interactive visualization or visual analytics has been studied - in visualization, designing collaborative visualization has focused on two different modes: synchronous and asynchronous collaborations. The major difference between synchronous and asynchronous collaborations is whether the users are co-located within the environment or not. Since it is difficult to understand how people share tacit knowledge within asynchronous collaborative visualizations, we limit our study to co-located collaboration environments such as a multi-touch surface (or sometimes referred to as a multi-touch table). In this section, we present a system that has been designed to answer how users share their tacit knowledge within such a collaborative environment.

Most visual analytics tools, with few exceptions, have been designed for single users in desktop environments. While often effective on their own, most single-user systems do not reflect the collaborative nature of solving real-world analytical tasks. Many intelligence analysts, for example, have been observed to switch repeatedly between

working alone and collaborating with members of a small team. In section 4.4.1, we propose that a complete visual analytical system designed for solving real-world tasks ought to have two integrated components: a single-user desktop system and a mirroring system suitable for a collaborative environment. To demonstrate our viewpoint, we adapt an existing single-user desktop analytical tool (iPCA) and convert it to a collaborative touch-table system (iPCA-CE) using a user-centric approach. Our two tools are designed to run concurrently, and are tightly integrated in that the findings and system parameters of one tool can be exported to the other. Using our two-pronged approach, an analyst can carry out an analytical task while switching between single-user and collaborative modes without losing track of the analysis process.

During evaluations with iPCA-CE, we noted that the system does facilitate the users in sharing ideas or knowledge with others. In section 4.4.2, we present the details of the evaluation, which involves observing 12 participants using iPCA-CE in order to understand their knowledge sharing behaviors within a collaborative touch-table environment. From the study, we noticed that their behaviors are different when solving analytical questions within three different conditions: single, double, and multiple.

In the following sections, we present both the iPCA-CE system, as well as the evaluations performed using iPCA-CE to understand the users' knowledge sharing behaviors.

4.4.1 A Continuous Analysis Process Between Desktop And Collaborative Visual Analytics Environments

In real-world analysis, domain experts often work together to solve analytical problems in a collaborative setting [104]. It is generally accepted that collaboration is beneficial for complicated tasks, and in the context of information visualization, groups working together have been shown to solve problems faster and more accurately than working alone [88]. However, it has been observed that real-world analysts typically perform both individual and group tasks, and as a result must frequently transition between single-user and multi-user collaborative workflows during the course of their analysis [28, 51]. Despite this fact, most visual analytics solutions have been designed either as standalone single-user applications or as purely collaborative systems, and very few analytical tools have been developed to cohesively support both activities.

We designed a complete visual analytical system for solving real-world tasks ought to have two integrated components: a single-user desktop system and a mirroring system suitable for a collaborative environment. To demonstrate our viewpoint, we adapted an existing single-user desktop analytical tool for exploring data using principal component analysis (iPCA) into a collaborative touch-table system (iPCA-CE) using a user-centric approach. The system parameters and analytical findings for these tools are tightly integrated so that analysts may seamlessly transition back and forth between single-user and collaborative work environments without losing track of the analysis process.

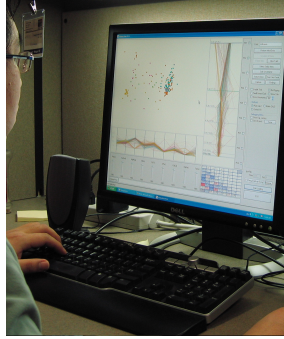
Although data analysis is often considered to be a stand-alone task, previous re-

search has shown that analysis of empirical data in collaborative environments is important and should be considered when developing visualization applications [24, 49]. While collaborative analytics can occur in a variety of interaction modalities, we focus specifically on collaboration using a multi-touch table based on existing work that has demonstrated potential increases in analysis performance [69]. Additionally, it has been observed in a case study that many experts consider collaboration to be a process for sharing the results and findings of analyses [28]. To support the sharing of analytical results, we defined an XML-based format for managing the findings from analyses, which can be exported when the user transitions between the single-user and collaborative environments.

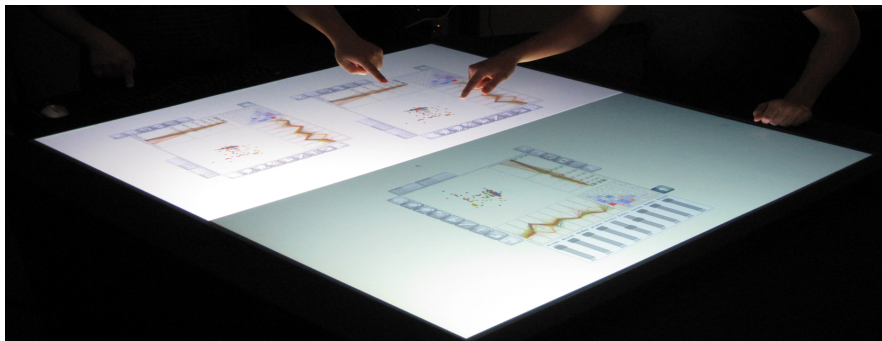
Converting an existing desktop application into a collaborative touch-table environment introduces unique technical challenges. The inherent differences between mouse and touch-based interaction require that the user interface must be redesigned, and limitations may also be imposed by factors such as system performance and screen resolution. However, despite the costs of converting an existing application into a new interaction modality, we believe that supporting both single-user and collaborative work in an integrated fashion provides important benefits for real-world analysis.

4.4.1.1 Related Work

While much notable research have been done for collaborative visualizations, studies pertaining to design and understanding for collaborative visual analytics has been limited. To the best of our knowledge, there has not been any integrated visual analytics system that supports switching between single-user and collaborative en-



(a)



(b)

Figure 44: (a) A single user performing an analysis of the Glass dataset (214×9 matrix) using iPCA. (b) Multiple users interactively collaborating on a multi-touch table using iPCA-CE.

vironments. Cambiera is a visual analytics solution which partially addresses these concerns and supports collaborative searching through large text document collections on a touch surface [69]. In addition to searching through documents, this system is capable of tracking the findings from analyses and maintaining awareness of collaborators' work. However, while Cambiera might support both environments, it is mainly designed for table-top collaborations and does not consider allowing analysts to migrate their findings between collaborative and single-user contexts.

In the community of computer-supported cooperative work (CSCW), numerous studies have been done. As a part of CSCW, building collaborative visualizations has a long history. Recently, Grimstead et al. [60] reviewed 42 collaborative visualization

systems in terms of five attributes: number of simultaneous users, user access control, communication architecture, type of transmitted data, and user synchronization. However, most of existing collaborative visualization systems are designed as asynchronous systems because a synchronous system is still limited in that people have to be in front of computers at the same time. Because of this limitation, Marchese et al. [87] emphasize the benefits of having an asynchronous collaborative system.

However, in visual analytics, a few studies have been done. Most of the studies have been performed on top of a touch surface. Cambiera (mentioned above) is an example supports collaborative search through large text document collections on a touch surface. The system is designed by considering collaborative activities to involve not just searching through documents, but also building individual's findings and maintaining awareness of another person's work. In visual analytics, it is still unclear how these systems should be designed, though guidelines have been suggested by researchers. Heer and Agrawala have provided design considerations for asynchronous collaboration in visual analytics environments [62]. However, to the best of our knowledge, there is no study to extend analytic processes across both single-user and collaborative modalities.

4.4.1.2 Converting a VA Tool From Desktop to Tabletop

Although supporting collaboration when solving real-world analytical tasks is important, most visual analytics tools, with few exceptions, have been designed as single-user desktop systems. Several researchers performed studies by adopting existing desktop visualization systems in collaborative environments [88, 67]. We believe

that they chose existing visualization systems because user-friendly visualizations in a collaborative environment enable users to find results more accurately. In here, we chose to extend a known and useful application to work in a collaborative touch-table environment. Multi-touch surfaces support a rich set of interactions that allow multiple users to work together to solve complex analytical problems interactively. We selected the Interactive Principal Component Analysis (iPCA) application, which has been shown to be an easy and effective desktop visualization for analyzing data sets and interactively exploring the parameters of principal component analysis [72]. Adopting a user-centric approach, we developed iPCA-CE, a complementary collaborative visualization designed specifically for multi-user interaction on a touch surface.

Together, iPCA and iPCA-CE form an integrated toolset which allows analysts to switch back and forth between the two visualizations on separate hardware without losing track of their current analysis tasks. While a single-user analysis could technically be performed on the touch-table using the collaborative application, this would not be as effective and productive as using the standalone desktop application. Since experts often prefer to work alone and switch their analysis process into a collaborative group activity only when necessary [51], it is important to provide both applications using hardware appropriate for the type of interaction required.

While iPCA is designed for use on a standard desktop computer, we deployed iPCA-CE on a multi-touch display system designed at the Renaissance Computing Institute (RENCI) [2]. It provides a 42" \times 46" work surface using two high resolution projection displays (2160 \times 1920 total). To notify the client system of input events, the table uses a multi-touch engine which detects finger touches and transmits events

using a networking protocol. Figure 44 shows examples of a single user performing an analysis with iPCA (Figure 44(a)) and multiple users collaborating with iPCA-CE (Figure 44(b)). These examples show the analysis of the Glass dataset, which is a publicly available scientific result from the UCI Machine Learning Repository [10].

4.4.1.2.1 User-Centric Approach

To convert iPCA into a collaborative touch-surface environment, we adopted a user-centric approach. Based on our observations of user behaviors when solving complex analytical problems, along with studies of real-world analysts [28, 51], we developed an informal model for the general analytical process that occurs as experts switch back and forth between single-user and collaborative workflows. Figure 45 shows an illustration of this process. In a desktop environment, a user performs a data analysis and compiles a list of results. When enough interesting results are found, the user meets with other experts in a collaborative setting to discuss the findings. Users share their findings from individual analyses with each other, and then work together interactively to perform a group analysis. Afterwards, the users take the findings from the collaborative analysis back into a single-user setting and then repeat the process. Thus, we designed our system so that users can continuously export their findings and system parameters back and forth between iPCA and iPCA-CE as they transition between single-user and collaborative contexts.

4.4.1.2.2 System Design

iPCA was designed with two goals in mind: (1) to help the user understand the complicated black box operation of principal component analysis [74] and (2) to allow

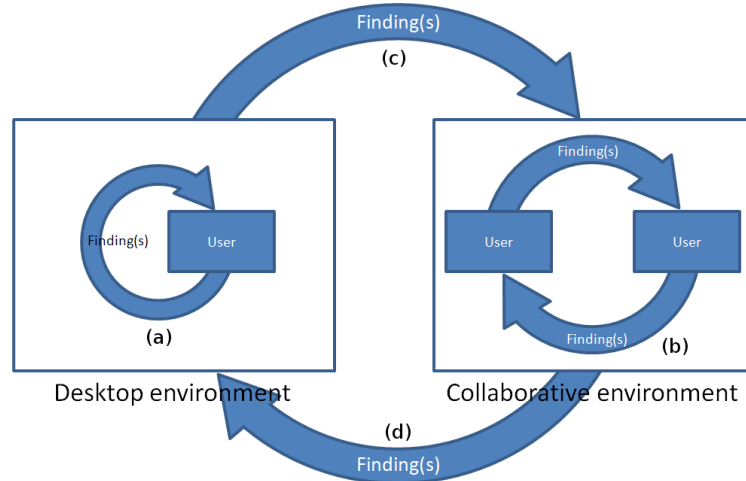


Figure 45: An illustration of users' processes of solving complex analytical problems and sharing analysis results (findings) within and between desktop and collaborative environments.

















the user to analyze complex data sets interactively [72]. We developed iPCA-CE to support all of the original features provided by iPCA.

Figure 46 shows the two systems. The user interface for both applications was developed using OpenGL. Similar to the original iPCA, the extended application (iPCA-CE) consists of four views: Projection view (Figure 46(a-1), Eigenvector view (Figure 46(a-2), Data view (Figure 46(a-3), and Correlation view (Figure 46(a-4)). In the Projection view, all data items are projected based on the first and second principal components by default. The Eigenvector view displays the calculated eigenvectors and eigenvalues in a vertically projected parallel coordinate view. The distances between the eigenvectors in the parallel coordinate view vary based on their eigenvalues, separating the eigenvectors based on their mathematical weights. The Data view shows the original data points in parallel coordinates. The Correlation view represents Pearson-correlation coefficients and relationships between variables as a matrix of scatter plots and values. All views are closely connected, such that if the

user changes the elements in one view, its corresponding results are updated in other views (*brushing & linking*). This interactivity allows the user to infer relationships between the coordinated spaces (see section 4.1.2 for detail).

A total of 18 touchable buttons were designed for interaction in iPCA-CE: nine buttons for interacting with represented data items (Figure 44(b-6)), six buttons for controlling the application (Figure 44(b-7)), one toggle button (Figure 44(b-8)) for expanding and collapsing the sliderbars panel, and two tab buttons (Figure 44(b-9)) for managing annotations and findings. Table 4 shows the touchable buttons and their meanings.

Table 4: Touchable buttons and their meanings.

Button	Meaning	Button	Meaning
	Go back to the initial state		Delete the selected item(s)
	Individual item selection		Partition the selected item(s) into a new workspace
	Range item(s) selection		Close the workspace
	Manipulation		Create a new workspace
	Toggle trails		Rotate the workspace
	Drawing annotation		Text annotation
	Switch between projection and correlation view		Capture current workspace as a finding
	Cancel the selected item(s)		Make the sidebar panel expand / collapse

4.4.1.3 Sharing Analytical Processes

In a single-user desktop environment, it is difficult to share analysis results with others. Consequently, for visual analytics systems, managing users' findings is as important as providing useful analysis tools because their end goal is to discover evidence that supports their hypotheses.

4.4.1.3.1 Managing Findings

A finding from visual analysis procedures may be represented as either a screenshot, which shows what was found during the analysis, or as an annotation, which explains in more detail what the screenshot represents. In iPCA and iPCA-CE, both screenshots and annotations are used to manage users' findings. The applications provide two methods for providing annotations: text and drawing. Text-annotation is an indirect approach for explaining the details of a user's finding. Drawing-annotation allows users to directly indicate important elements or features visually on screen. In iPCA, both methods of annotation are performed using a keyboard and mouse.

However, in iPCA-CE, annotations needed to be supported differently because all interactions are initiated by finger touches. Therefore, a virtual keyboard is displayed for text-annotation, and a drawing tool is used for drawing-annotation. Figure 47 shows examples in which users try to describe what they found in iPCA and iPCA-CE.

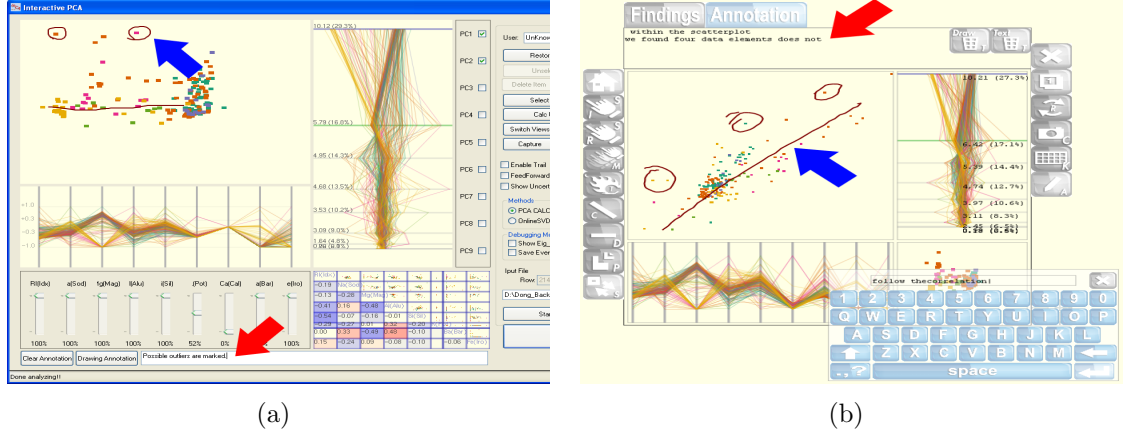


Figure 47: Annotation examples in (a) iPCA and (b) iPCA-CE. Text-annotation (red arrows) contains data inputted using either a desktop keyboard (iPCA) or a virtual keyboard (iPCA-CE). Drawing-annotation (blue arrows) allows users to directly annotate onto the display using a drawing tool controlled by a mouse (iPCA) or finger touches (iPCA-CE) .

Findings are stored in an XML format similar to the P-Set model [70]. However, since our design philosophy does not require us to track all of a user's exploration procedures, we only describe each finding with parameter sets. The parameter sets are similar to the sets defined in the P-Set model, though defined specifically for our visual analytics system. Sets represent interactive operations (such as selection and deletion), view, sliderbar control, text- and drawing-annotations, and a final result. Since users' findings can easily be recreated with the parameter sets, it is an important and useful feature for sharing findings with others. Each finding is identified by the

user's name and a timestamp. Figure 48 represents an example of a finding definition (Figure 48(a)) and a screenshot of the final result (Figure 48(b)).

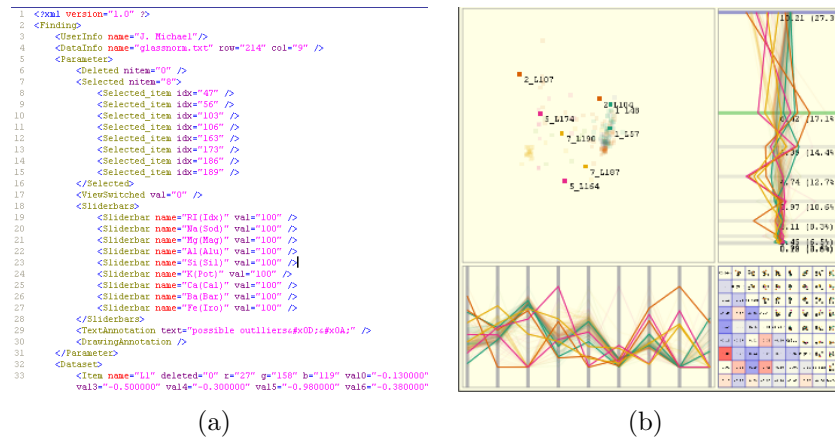


Figure 48: An example of (a) an analytical finding defined in an XML format and (b) a screenshot of the final result. A user has selected eight possible outliers during the analysis, which are highlighted in the screenshot.

Figure 49 shows how findings are managed in the two systems. In iPCA, all findings are listed chronologically in a separate window, and the user is provided with buttons for updating or deleting findings in the list 49(a). A finding is created using the current view and annotations and added to the list by selecting a button in the main window. In iPCA-CE, however, the differences in display and interaction require that findings be managed differently than the desktop applications. Findings are managed within a tabbed window activated by a button above each workspace 49(b). Each finding is represented as a screenshot thumbnail identified with the user's name and timestamp. Findings are created by touching a capture button in the workspace (see Table 4), and findings can be moved into the workspace for updating via a simple drag-and-drop operation.

In iPCA, the user can continuously create findings and track the history of the

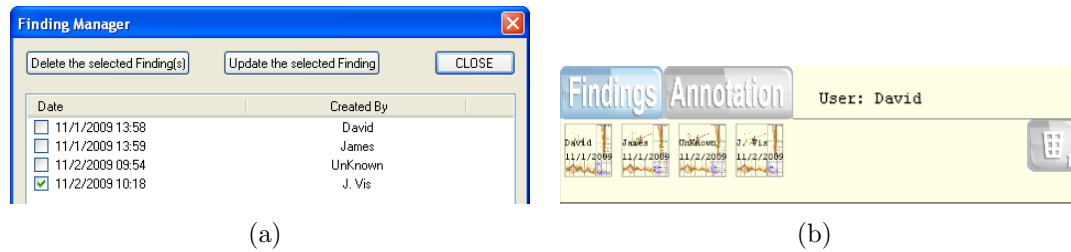


Figure 49: Management of findings in (a) iPCA and (b) iPCA-CE. Findings are displayed chronologically and are identified by a timestamp and the creator's name.

analysis by viewing the previously created findings (see Figure 50(a)). If the user wants to share with collaborators, the XML file can be transmitted to others and imported into their applications. In iPCA-CE, a finding can be directly transferred to others by dragging it into their workspaces (see Figure 50(b)). Using this direct passing operation, collaborators can easily become aware of each others' analytical processes and results, although they each still maintain an individual workspace for performing their analyses. Since this sharing operation should be subject to the agreement of the collaborator, a confirmation window is displayed to ask for permission to accept the finding being shared by another. Furthermore, the findings from iPCA-CE can also be imported into iPCA. This feature is especially important when a user leaves the collaborative environment and wants to continue the analysis in the desktop application.

4.4.1.3.2 Sharing Findings

As illustrated in Figure 45, analysts may perform four distinct sharing processes: (1) asynchronous self-sharing in the desktop environment, (2) synchronous sharing in a collaborative environment, and (3 and 4) two asynchronous transitional sharing processes between the desktop and collaborative environments. All four sharing

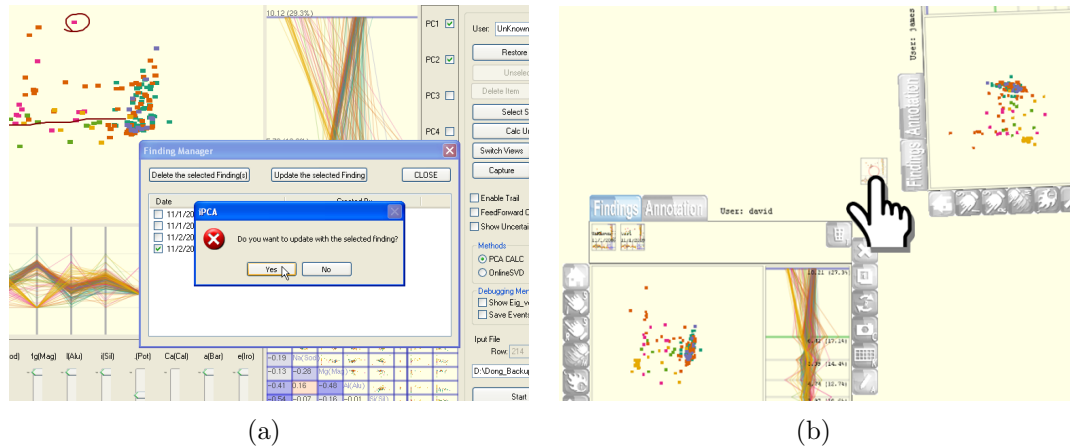


Figure 50: (a) Updating the workspace with a previously created finding in iPCA. (b) The user performs a drag & drop operation to pass a finding from the original workspace (bottom left) to the other collaborator's workspace (top right) in iPCA-CE.

processes can be supported by passing the finding parameters between users and applications. Processes (1) and (2) form continuous loops within each environment, while processes (3) and (4) form a global loop through which the entire analysis process iterates over time. By providing support for all four sharing processes, we form an integrated visual analytics system which reflects the analytical processes carried out by real-world experts.

(1) Asynchronous self-sharing occurs in most single user desktop applications (Figure 45(a)). Previously saved findings can be displayed, which allows the user to track the history of the analysis and continuously update previous findings based on recent results. (2) Synchronous sharing in the collaborative environment provides users with the capability of viewing others' findings to gain understanding of collaborators' analytical processes (Figure 45(b)). This type of sharing allows users to work together simultaneously to find new analytical results. (3) Transitional sharing from the desktop application to the collaborative environment represents the process by

which the results from a single user's analysis are made public for the rest of the group (Figure 45(c)). This is analogous to traditional procedures for sharing analysis results, such as preparing presentation slides or written reports to present and discuss in a group meeting. (4) Transitional sharing from the collaborative environment back to the desktop application has not been considered in many previous collaborative visual analytics systems (Figure 45(d)). However, we believe this is an important method for users to further investigate interesting results that were shared during the collaborative session.

4.4.1.4 Challenges

The process of converting a visual analytical tool from a desktop application (iPCA) into a collaborative touch-table environment (iPCA-CE) introduces many unique technical challenges. In this section, we explain the design issues that we considered when developing the collaborative application.

Performance Although Piringer et al. [105] emphasize the importance of adopting multi-threaded operation to control both continuous interactions and a massive amount of data simultaneously, the multi-threaded operation is not always required for desktop-based visual analytics applications since all interactions are based on single mouse input. However, in a collaborative environment, threading is necessary to manage the display and listen for incoming touch events.

Since iPCA was designed for operation on a desktop, the application was not multi-threaded. However, as mentioned previously, the iPCA-CE application was designed to support collaborative visual analysis using a multi-touch engine developed

by RENCİ [2], which detects finger touches and sends input event messages to the client system using a networking protocol. As a result, it was necessary to design iPCA-CE as a multi-threaded application to process all incoming touch messages and initiate relevant interactions.

Unlike iPCA, which contains a single workspace, iPCA-CE supports creating multiple workspaces simultaneously. We followed a workspace-based design approach for this application, with each workspace operating independently. Touch messages are filtered based on the currently open workspaces and are not processed outside of these areas. This technique is useful in supporting efficient simultaneous processing of multiple touch inputs over multiple workspaces.

Interface iPCA supports mouse-based interactions to change parameters, select or remove data items, and navigate within the display space. However, since mouse-based interaction is not available in a touch-surface environment, these functionalities need to be redesigned to work with finger-touch interactions. Since many interactions were originally intended to be initiated by mouse hovering and button clicking, which are not possible on the touch-surface, this poses a notable challenge in providing the same functionality as the desktop application. Many interactions on the touch-surface were accomplished by pressing touchable buttons which implemented the same functionality for the mouse-based interactions as well as the static menus and icons from the desktop application.

Some operations on the touch-surface were easily mapped to touchable buttons, such as selection and deletion. The zooming operation, however, leverages the unique

capabilities of the multi-touch surface, and is activated by moving two finger-touches closer (zoom-in) or farther apart (zoom-out). The panning operation is initiated by dragging a finger on the surface. However, it is difficult to provide intuitive support for the rotation option simultaneously, so this functionality is activated by enabling a touchable button and then controlled by finger dragging.

Screen Resolution As mentioned previously, we deployed iPCA-CE on a touch-surface which provides a high-resolution display (2160×1920). The resolutions of desktop environments vary widely, and usually do not provide such a high resolution using a single display. Since iPCA is designed to work on a 19" monitor with a resolution of 1280×1024 or less, a direct conversion of the desktop application on the tabletop is not feasible. Despite the high resolution, screen real estate is relatively limited for the collaborative workspaces, so by default the sliders are hidden from view, and can be displayed by touching the expand/collapse button (see Table 4). Similarly, the tab regions for displaying findings as well as text annotations can be expanded and collapsed.

In iPCA, mouse-based interactions allow precise selection of data elements. In contrast to the desktop, precise finger input using a touch-surface is comparatively difficult since users might have difficulty performing selection operations if a button is smaller than their fingertips. To address this, the size of each touchable button needed to be large enough to support convenient use. This is a common problem for touch-based systems, especially handheld devices. However, the high-resolution display provided by our touch-surface makes it possible enforce the minimum size for

each button to be the size of an average fingertip.

4.4.1.5 Discussion

We believe that converting iPCA from a single-user desktop analytical tool to a collaborative touch-table system is important and provides many advantages. However, since there is no one-to-one mapping between desktop-based systems and collaborative touch-table systems, the benefits of this approach need to be balanced against the costs of the conversion process.

Benefits The most promising benefit of developing both a single-user desktop application and a multi-user collaborative system is that these tools more accurately reflect the analytical processes carried out by real world experts. Analysts can transition between single-user and collaborative workflows seamlessly without losing track of their current analysis. In the desktop application, users' findings are described as parameter sets which can be used for history management in a single-user context or transmitted to others for sharing analysis results. Users can move their findings back and forth between the desktop application and the collaborative environment to perform group analyses. Additionally, in the collaborative environment, users can intuitively share their findings with others using direct drag & drop operations between multiple workspaces.

Costs Ideally, the desktop application and collaborative environment should mirror each other as closely as possible in terms of visual representation, user interface, and functionality. However, due to inherent differences between desktop and touch-table environments, it is impossible for the two applications to be purely identical. Specif-

ically, the user interface for a desktop application will need to be altered to support touch-based interaction instead of mouse-based interaction. Since a standard mouse supports 2D motion as well as multiple buttons, mapping mouse-based interaction to finger touches is a complicated procedure. Also, touch-based interaction is not precise as mouse input, so the size of touchable components needs to be carefully adjusted. Additionally, in desktop environments, menu systems are commonly used to manipulate application system parameters. However, menus are difficult to implement in a touch-table environment due to the lack of input precision, so the interface for this functionality also has to be redesigned. Since screen real estate on a touch-surface is limited, designers may want to consider including only the important and necessary functionality from the desktop application in the collaborative environment.

Although many useful visual analysis applications have been developed to assist users in understanding complicated relationships in large data sets, they are mostly limited desktop applications designed for single-users. Collaborative visual analytics environments have also been developed, which allow users to work together to solve complex analytical problems. However, on their own, neither of these two environments accurately reflect the continuous analytic processes carried out by real world experts. In this section, we converted a previously developed single-user desktop analytical tool to a collaborative touch-table environment using a user-centric approach. The two tools are designed to run concurrently, and are tightly integrated in that findings and system parameters of one tool can be exported to the other. Using this approach, users can continuously carry out analytical tasks while switching between single-user and collaborative modes without losing track of the analysis process.

4.4.2 Understanding Users' Knowledge Sharing Process

Understanding how people act when analyzing data is an important research topic in visual analytics, but it is also extremely challenging [137]. In order to isolate the problem, we focus our research on the understanding of analytical behavior to be strictly within the context of a collaborative environment, specifically on a touch table. Our work begins with understanding collaborative analysis procedures, which attempts to understand users' knowledge sharing process by observing 12 participants using the iPCA-CE system.

4.4.2.1 Understanding Collaborative Analysis Procedures

In order to design our study procedures, we first need to understand the nature of the collaborative analysis procedures. While a collaboration process can occur through the use of collaborative visual environments, the most natural method for sharing tacit knowledge is still direct verbal communication between users. Through direct communication, the users actively share their discoveries and tacit knowledge and incorporate each other's domain expertise into their own. By reviewing existing literatures, we classifies the users' collaborative analysis procedures based on the number of tasks and workspaces (Table 5).

Table 5: Collaborative analysis procedures for multiple users in collaborative environments. Based on existing literature, synchronous analysis procedures and their characteristics are classified in terms of the number of task(s) and workspace(s).

Task(s)	Workspace(s)	Cooperative	Coupling	Example(s)
Single	Single	Joint	Tightly coupled	[67, 88]
Single	Multiple	Joint	Mostly Loosely coupled	[69, 88]
Multiple	Single	Disjoint	Loosely coupled	[67, 130, 142]
Multiple	Multiple	Joint or Disjoint	Tightly or Loosely coupled	[20, 120]

The term “cooperative” indicates whether users collaborate with each other to solve given data analysis tasks. If users solve a task together, collaboration between them might take place to increase the overall performance of solving the task (joint). If users solve different tasks, it is not necessary to have collaboration between them (disjoint). The “coupling” refers to the dependency of users on one another to make significant progress [130]. If users need to interact frequently relative to the amount of work to be done, the work is tightly coupled; conversely, when users have few interactions or independently work for long period of time, the work is loosely coupled [115, 130].

A single task and a single workspace If multiple users work on solving the same task within a co-located workspace (a single workspace), it has been shown that the users prefer to collaborate with each other synchronously to solve the given task. However, if one person manipulates the display, the others would simply observe the display to build new ideas or strategies [130, 142].

A single task and multiple workspaces If each person has her own workspace to solve the same given task (multiple workspaces), people tend not to collaborate synchronously with each other. Instead, they prefer to collaborate with others only when it is necessary. This is because when given individual workspaces, each user would work mostly within their own workspace to build their understanding. Only after they have gained some understanding of the problem would they begin to collaborate with the others to share their findings and strategies. In this scenario, it might be useful to have a separate shared workspace to facilitate the communication of independent findings [68].

Multiple tasks and a single workspace If the users are trying to perform multiple tasks within a single workspace, visual awareness [142] of what other users are doing within the workspace should be considered and managed. Users begin with their own working regions within the shared workspace, but these regions can change and overlap if their regions of interest are growing. In such case, unwanted interference might occur. To solve such problems, Tang et al. [130] emphasize that fluid transitions between coupling styles should be considered when designing collaborative environments.

Multiple tasks and multiple workspaces Having multiple tasks and multiple workspaces is a common approach within asynchronous collaborative environments [62]. However, limited study has been done on multiple tasks and multiple workspaces within synchronous collaborative environments. Scott et al. [120] observed tabletop collaboration in casual and formal settings and found participants would intuitively divide their workspaces into three general types of interaction areas: personal, group, and storage. People spend most of their time working within personal workspaces and move to group or shared workspaces when they collaborate each other on the main tasks that contribute toward the overall group's goals. Also Butkiewicz et al. [20] designed a probe-based geospatial collaborative environment having three types of interaction areas similar to Scott et al.'s table collaborative environment. Since their environment uses geospatial datasets, the group workspace (a map) is positioned in the center area of the environment, within which users share their regions of interest. Users privately analyze the contents of their regions of interest within their personal

spaces. The periphery of the environment, i.e., areas along the edges and out of users' easy reach, becomes a storage space for the result of completed analysis. To control and manage different workspaces, they adopt probe interfaces that monitor and keep track of each user's analysis.

4.4.2.2 Conducting a Study

Based on our understanding of the characteristics of collaborative data analysis procedures, we observed that all four analysis procedures are often considered in collaborative visual analytics applications. In our study, we extend the characteristics of collaborative analysis procedures to include three conditions. The three conditions are designed based on the number of workspaces as single, double, and multiple workspaces. In all conditions, two participants are requested to work together to solve a given task. The single workspace condition dictates that two participants can have only one workspace. The double workspace condition allows each participant to have her own workspace. In the multiple workspace condition, the participants are allowed to create as many workspaces as they want.

In the study, a collaborative visual analytics application (iPCA-CE) is used (see section 4.4.1 for detail). Since the purpose of this study is to understand how participants collaborate together and share knowledge in a collaborative environment, all their interactions on a multi-touch table are screen-captured internally and externally. Internally, the screen is captured using a screen capturing software. Externally, the screen is captured using a high-definition camera facing down from the ceiling. Also their verbal communications are audio-recorded. Additionally, all participants'

time-stamped interactions are captured by built-in functions of the system and saved automatically into log files.

4.4.2.2.1 Procedure

12 participants (nine males and three females) participated in the study. All participants are graduate students in computer science. Most participants self-reported that they were familiar with visualization applications. About four participants claimed that they do not have an experience of using a multi-touch table.

All participants were administered a questionnaire containing demographic questions about gender, major, and familiarities on visualization applications and a multi-touch table. Before beginning the each of conditions, which were counterbalanced to control for order effects, participants were given a tutorial about the tool they were about to use, including how to utilize basic functionalities. Sufficient time is given to each participant to be familiarized with each scenario and the visual analytics application on the multi-touch table. Any questions the participant had were answered. After the training, they were requested to solve a task question in each condition.

We performed the evaluation using four different datasets: the Iris dataset (150 data items \times 4 dimensions), the E.Coli dataset (336 data items \times 7 dimensions), the Forest Fire dataset (517 data items \times 11 dimensions) and the the Glass dataset (214 data items \times 9 dimensions). The Iris dataset was used in the training session whereas the E.Coli, the Forest Fire, and the Glass datasets were used in the actual evaluation. All categorical variables in each dataset are removed for PCA calculation. All four datasets are scientific results that are publicly available at the UCI Machine Learning

Repository [10]. All datasets are also counterbalanced.

The used task question is to find the most striking outlier(s). All participants were requested to have a discussion with their colleagues to justify their findings. At the same time, they were also requested to find evidence for their findings and provide both their final answers and evidences by using both text and drawing annotation methods (see section 4.4.1.3 for detail about annotation methods).

All participants' time-stamped interactions were captured and saved into log-files such that several variables relating to the use of the visualization were tracked. These variables include completion time and time-duration for each touch interaction. The number of touches are also tracked. Three types of interactions were tracked: basic manipulation interactions (zooming, panning, and rotation), touchable-button interaction, and annotation interactions (text- and drawing-annotations). As soon as each condition is completed, a post-interaction questionnaire was administered to evaluate how easy and intuitive the visualization is for collaborating with the other. These questions include "how easy or difficult was this method to share your ideas?," "how intuitive was this method to use?," "how well did you communicate with your partner?," and "what aspects of the interaction made using the visualization more difficult to use for collaborating with a colleague?" After a participant completed the evaluation of all conditions, a post-study questionnaire was administered, which requested feedback on the comparative ease of use of all conditions, including open-response questions about the participants' likes and dislikes.

4.4.2.2.2 Evaluation Results

We present the results of our evaluation based on accuracy, easiness & intuitiveness, usefulness, effectiveness, and preference. Time is measured quantitatively; whereas the other categories are analyzed based on the participants' qualitative feedback.

Accuracy Figure 51(a) shows the results of the participants' completion time and verbal communication time while solving the task question. As shown, the participants spent about 531.57 seconds on average to find the answer for the task question in each condition. While solving the task question, they spent an average of 211.64 seconds to verbally communicate with each other to discuss their ideas or to justify their findings. On average, they found about 4.5 findings in each condition (Figure 51(b)). Based on the statistical analysis (one-way ANOVA), we find that the completion time ($p = 0.28$) and the verbal communication time ($p = 0.86$) are not statistically significant across the condition. However, we find that the number of findings across the condition is significant ($F(2, 17) = 5.65, p = 0.01$). As shown in Figure 51, the participants spent less time to solve the task question in the single condition. However, they spent more time to communicate each other in the single condition. Since the participants prefer to discuss about their ideas and findings, their overall findings are a bit less than the other two conditions.

By measuring how much time the participants spent to interact with the system (iPCA-CE), we find that the total time spent to perform interactions ($p = 0.54$) is not statistically significant across the conditions. Also the time-spent for the touchable-button interaction ($p = 0.17$), the basic manipulation interactions ($p = 0.07$), and the

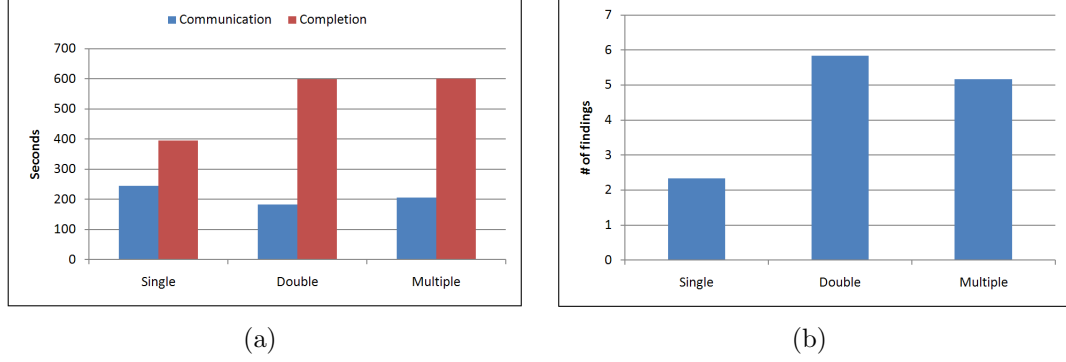


Figure 51: Average completion time and verbal communication time (a) and the number of findings (b) across the condition.

annotation interactions ($p = 0.52$ for drawing-annotation interaction and $p = 0.73$ for text-annotation interaction) are not significant. Although we assumed that the participants might spend less time interacting in the single workspace condition, these results indicate that the participants spent similar amount of time to perform each interaction (Table 6).

Table 6: Average time-spent (seconds) to perform each interaction.

	Manipulation	Finger-touch	Touchable-button	Drawing-annotation	Text-annotation
Single	17.43	91.27	8.22	13.11	25.64
Double	16.84	154.60	10.06	16.92	48.90
Multiple	29.68	118.34	7.94	10.78	41.58

Easiness & intuitiveness (post-condition questionnaire) The participants were requested to report the easiness and intuitiveness of the condition in solving the task and communicating with their colleagues.

Figure 52(a) shows that most participants (86%) reported all three conditions to be “easy” or “very easy” when solving the task question. Similarly, about 80% of the participants identified the conditions as being “intuitive” or “very intuitive” to solve the task question. In addition, the participants were requested to answer the

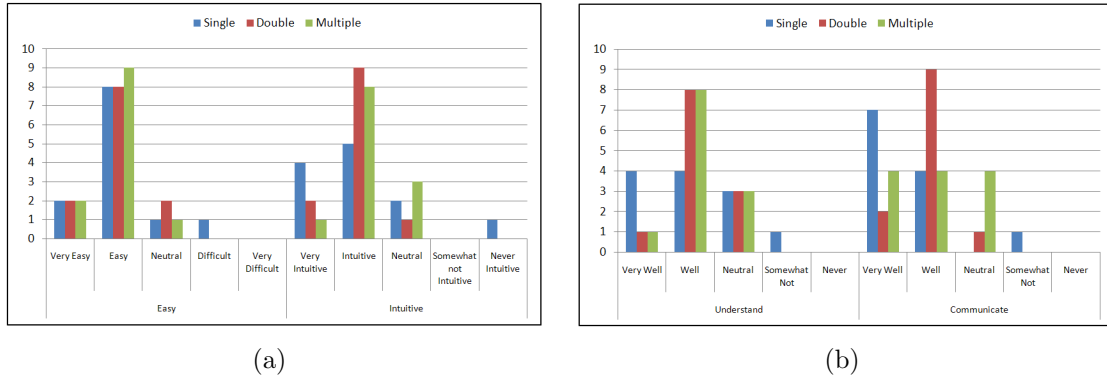


Figure 52: Four study results. (a) Ease of use and intuitiveness of use for each condition, and (b) “how well did you understand the data?” and “how well did you communicate with your colleague?” are measured.

questions “how well did you feel like you understand the data?” and “how well did you communicate with your partner?” About 83% of the participants indicated that they communicated with their colleagues to share ideas “well” or “very well”. And about 72% of the participants mentioned that they understood about the data they used “well” or “very well” (Figure 52(b)).

Interestingly, as shown in Figure 52, a participant indicated that the single workspace condition made it difficult for her to use and communicate with her colleague because they had contradictory ideas about the analysis procedure.

Preference (post-study questionnaire) After the evaluation, each participant ranked the three conditions, provided their preference, and described their pros and cons.

Based on the description of the pros and cons, we find that most participants pointed out the multi-touch table interactions as quite helpful to understand the data. About three participants complained about the false-touches and the system delay. Overall, the feedback we received for the collaborative visual analytics application was very positive. All participants found the tool to be useful and intuitive to use,

and efficient for collaborating with each other.

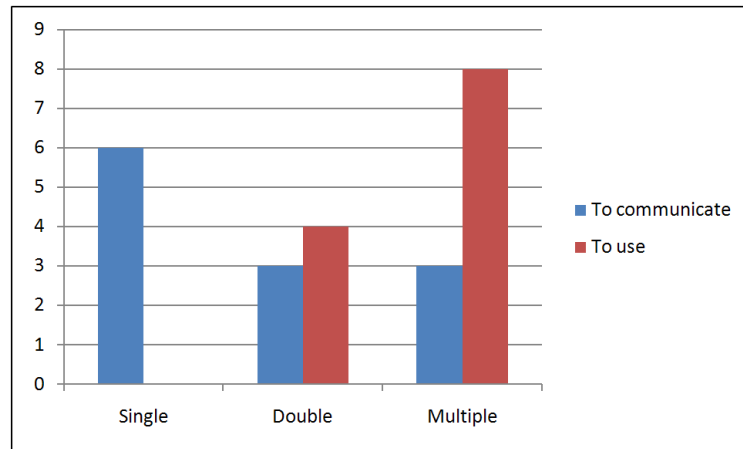


Figure 53: The participants' preference to use and to communicate with others.

In regards to the participants' preference (Figure 53), we find that the participants preferred to have the multiple workspaces for solving problems. Interestingly, about a half of the participants preferred having a single workspace for communicating and sharing ideas with others.

4.4.2.3 Discussion

From this study, we find that the participants continuously shared their ideas in the single workspace condition because they need to have an agreement before performing an action. In the single workspace condition, we find that interference [141] occurred between collaborators because their desired working areas often overlapped with the other's. We also observe that if one participant interacts with the display, the other participant simply observes what the other participant is doing and tries to create her own ideas or strategies. This pattern of collaboration alternated continuously as different dominant users interacted with the display. In the double and multiple workspace conditions, a participant pointed that he felt like he was racing with his

colleague to find many possible answers.

As mention above, Scott et al. [120] noticed that people prefer to have three general types of interaction areas: personal, group, and storage. However, in our study, we noticed that they do not create a group (or shared) workspace. Instead, they prefer to send their important findings directly from their workspaces to the other collaborator. We postulate that the difference in the results is due to the fact that in the study by Scott et al., the experiment was performed using a real table, which had divided sections. In our study, our visualization design on the multi-touch table did not have physically divided sections, which allowed the user to more directly interact with each others' workspaces.

CHAPTER 5: DESIGNING KNOWLEDGE VISUALIZATION

As we shown in Chapter 3, the knowledge visualization framework is designed having general properties of knowledge (explicit) and personal properties of knowledge (tacit) in visualization. With the knowledge visualization framework, several studies have been conducted to understand knowledge conversion processes in visualization (see Chapter 4). This chapter begins with providing a description how to design a knowledge visualization application.

With using the knowledge visualization framework, we integrated the concept of knowledge visualization to our existing visualization application called GVis [65]. GVis is developed to help bioinformaticians investigate, analyze, and understand large amounts of complex genomic data. The knowledge visualization framework we applied here is to make these difficult tasks easier by organizing and representing the knowledge associated with the genomic data. In this chapter, we explain in detail how we apply the knowledge visualization framework to the existing application and what improvements have been accomplished. We begin with explaining the initial version of GVis in terms of what features are supported and what visualization techniques are provided. Then, we move to explaining the updated genomic visualization with the knowledge visualization framework applied.

5.1 Design Issues

In Chapter 3, the knowledge visualization framework is provided. Since knowledge visualization mainly focuses on representing knowledge, it is important to consider finding important knowledge and representing them in visualization. Prior to finding the important knowledge, collections of explicit knowledge need to be managed by referencing existing explicit knowledge. Alternatively, explicit knowledge K_e can be extracted from data D based on some function f , the relationship between these variables can also be expressed as a set of equations:

$$K_e = f(D); I(t) = V(K_e, S, t) \quad (10)$$

Based on the explicit knowledge K_e , some specification S , an image at time t , $I(t)$, can be created through the visualization V . This image is then perceived and understood by the user, resulting in an increase of the user's tacit knowledge K_t .

While we believe the knowledge visualization model (shown in Figure 3) is accurate conceptually, in practice, we have come to realize that the amount of explicit knowledge K_e that is derived from a complex dataset D could be nearly infinite. The process of converting explicit knowledge into visualization is thus limited by the screen resolution and computer hardware and often cannot be displayed in its entirety. To this end, we refine the knowledge visualization model and subdivide explicit knowledge K_e into smaller collections of knowledge K_{e_1} to K_{e_n} (Figure 54). The challenge of creating a visualization therefore becomes the determination of the most relevant and valuable collection of knowledge to display to the user.

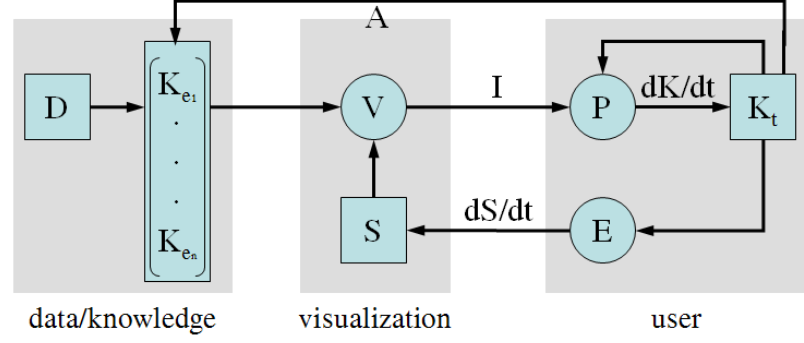


Figure 54: The refined model of the operational model in Figure 3 having the collections of explicit knowledge K_{e1} to K_{en} .

To design a knowledge visualization application, it is important to consider and follow the knowledge visualization framework. In addition, it is also necessary to support the four knowledge conversion processes (see Chapter 4) in visualization. To support the internalization process, it is important to provide a rich set of interactions. With such interactions, users can continuously internalize tacit knowledge by looking at and interacting with the visualization. Since their tacit knowledge can be externalized and used to design or redesign a current visualization representation, the visualization should have a capability of externalizing and integrating (reusing) their tacit knowledge into visualization. As we mentioned above, the collections of explicit knowledge can be created through the combining process. Although adopting existing knowledge is regarded as a common, but important approach to strengthening the visualization, combined knowledge should be carefully validated because inserting unrelated or incorrect knowledge could potentially degrade the value of visualization. Visualization designers need to validate the combined knowledge by working with domain experts. Lastly, the knowledge visualization application should be designed having an ability of supporting the collaboration process. Although there are several

issues need to be accomplished, a knowledge visualization application, in general, can be designed by representing important knowledge and supporting the knowledge conversion processes in visualization. In following sections, we provide an example of knowledge visualization application.

5.2 Genomic Visualization (GVis)

GVis was developed to support the visual analysis of large-scale phylogeny hierarchies populated with the genomic data of various organisms. It uses a publicly available biological database (GenBank) from NCBI to create the phylogeny hierarchy and allows the user to quickly browse the hierarchy from the highest level down to the level of an individual organism for the desired organism of interest. Roughly, the GenBank currently holds more than 40,000 genomic data and about 18,000 publication citations. GVis consists of several interlinked windows that support the user in visual exploration and detailed analysis of the genomic data without losing her focus as she deals with multiple levels of information hierarchies. Several properties provided in GVis include: (1) a multiscale structure based on the Pad++ “infinite pan and zoom” paradigm [11, 54], (2) a general tree structure capable of providing quick access to many thousands of organisms of any size, (3) a layout scheme for arranging genomic information, including annotations, at different scales that support iterative analysis and comparison, (4) level-of-detail techniques to continuously reveal increasing amounts of detail as one zooms in, (5) multiple interaction techniques for presenting “details on demand” through direct interaction with the visual presentation, and (6) integrated analysis tools that can be launched within GVis and whose

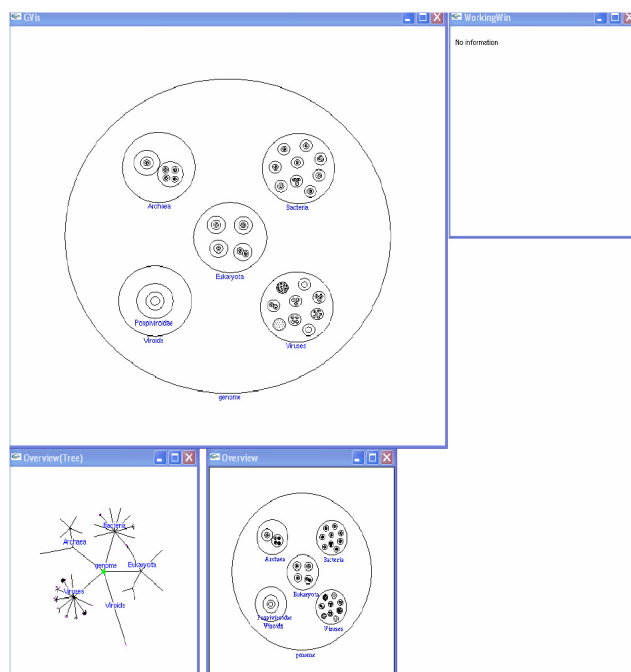


Figure 55: The overview of GVis. It has multiple windows: a zoomable window (top left), two overview windows (bottom), and a working window (top right). These windows are closely coordinated with each other to maintain interactivity within the visualization.

results can then be compared.

5.2.1 GVis Framework

Figure 55 shows the overview of GVis. The overall GVis framework consists of a main window, an overview window, an overview tree window, and a working window. The main window is a multi-resolution display of genomic information that supports interaction features such as zooming, panning, and various mouse events. The overview window is designed for showing the user's positional information because the user can easily lose his way in a scalable visualization space. The overview tree window serves a similar purpose as the overview window, but shows the overall layout using a circular tree structure. Finally the working window is designed for

comparing genomic data using a sequence analysis tool as described later. Although the main window, the overview window, and the overview tree window use different modes of interaction and visualization, they are tightly integrated. When the user changes focus in one of the windows, the other two will automatically change their viewing perspectives accordingly.

5.2.1.1 Representing the Phylogeny Structure

GVis builds a visualization structure based on a universal model containing comprehensive domain knowledge appropriate for genomic investigations. A universal structure is quite important because it can be used everywhere, on servers or by individual users collecting and annotating data, and is known a priori by everyone. This means that everyone knows where to put a new genomic annotation or find an existing piece of data. A universal structure means that genomic data can be highly distributed and easily shared. Everybody knows the structure but nobody possesses all the data. However, parts of the structure can easily be compared between users to coordinate their contents. We choose the comprehensive phylogeny tree for all organisms as the universal structure. This tree has a place for everything in its branches, including archaea, bacteria, viruses, and eukaryotes. The phylogeny tree is widely recognized by users. It tends to group genomic data with similar characteristics in size and functional structures on nearby branches. The phylogeny classification structure embodies how biologists tend to think about organisms. Nevertheless, it is also true that no structure, even if comprehensive, can encompass all the relationships that genomic scientists discover and investigate. Indeed, it is increasingly true

that scientist compare quite disparate species (e.g., viruses and bacteria) to discover common mechanisms. Finally, it is important to note that the data layout and organization mechanism does not depend on the details of the structure used. If a universal structure other than a phylogeny tree were deemed more useful, the data could be efficiently organized for this structure.

The embodiment of the phylogeny structure is a taxonomy tree based on the taxonomy structure defined in NCBI. In GVis, each node is represented by an ellipse and each leaf node is represented by a genomic structure as shown in Figure 55. GVis adopts a containment relationship between parent and children nodes, which is similar to a Venn Diagram. Figure 56 illustrates the conventional node link tree structure in this representation. The location and size of child nodes inside the parent node depends on the number of children nodes, with no child nodes overlapping. First, locations are assigned randomly around the origin and then each of the locations is considered as a particle that pushes others nearby. To prevent particles from moving too far away, we add a counter-force that attract particles towards the origin (and keeps them within the parent node). These two complementary forces move particles in equilibrium positions eventually, and we then store them.

When GVis loads the tree structure, depending on the number of children, it reads the location table and assigns this spatial coordinate into the node. In this way, it accelerates the loading process because the system does not need to compute the location of the children and the users can freely add or remove genomic data from a directory without worrying about changing the spatial information. Because of the containment relationships between parent and child nodes, the tree can be efficiently

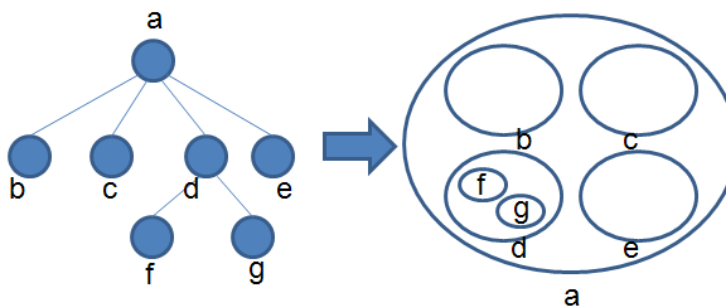


Figure 56: Translation from tree structure to Venn diagram. Each node in the tree structure (left) is represented by an ellipse (right).

traversed. When the root node is put into the visual query, GVis will decide if the root node needs to be drawn by checking whether the view window contains or intersects it. If so, GVis traverses to the next level of the tree and applies the same test recursively. At some level, the size of node on the screen will be too small compared to a screen-space threshold, and this is the point when the traversal stops. In this way, the tree can be traversed very quickly.

5.2.1.2 Interactions in GVis

Since the total size of the GVis gene database will grow significantly as users start acquiring more data of interest to view, we have developed data management techniques that apply a level of abstraction approach across the genomic data in the database. Clearly, there is no point in drawing millions of nucleotides per organism when we are viewing all of them at the broadest level of overview, since each nucleotide would project to a small fraction of a pixel. For efficient navigation and exploration, data should be stored in a spatial hierarchy for incremental access and multi-resolution display such as “Pad” [99] metaphor and its extension, Pad++ [11, 54]. In GVis, we have used the geographical layout to lay out this structure to satisfy the criterion that

the contents of children be totally contained spatially in their parents and not overlap each other (see Figure 57). This makes the domain-specific structure especially efficient to traverse and query spatially. The domain-specific structure encompasses both the 2D spatial extent and the scale dimension (through use of tree depth). It lends itself to interactive navigation of the scalable data space, with incremental updates from out-of-core data storage.

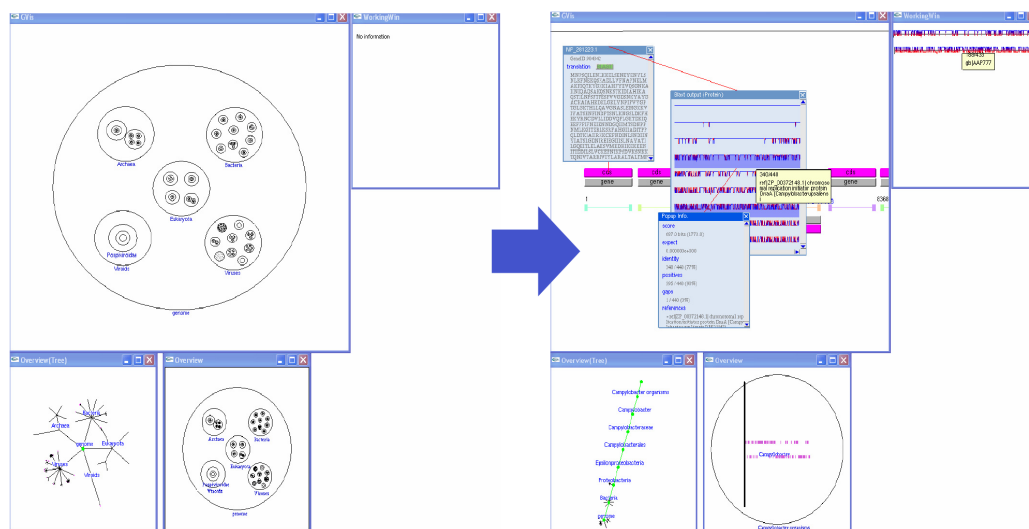


Figure 57: Zooming from root (all organisms) to a specific organism (Campylocater).

As mentioned above, GVis uses a collection of genomic information from NCBI. For efficiently displaying this information (annotations about each gene) on screen, a window-based approach is used. All the information is displayed and managed in an annotation window (also called inner-window). In addition to displaying text annotations, an inner-window also allows launching of sequence analysis tool through clickable buttons and selection of gene sequences through mouse click-and-drag. A detailed explanation about the gene sequence analysis in GVis is given in the following section.

5.2.1.3 Performing Sequence Analysis

GVis does not only represent the genomic information, but also supports interactive gene sequence analysis. It integrates a network-based gene sequence matching tool by NCBI called netBLAST. netBLAST is a publicly available gene sequence matching program that emphasizes regions of local alignments in order to detect relationships among sequences that share isolated regions of similarities. In the inner-window, a user can select an arbitrary length of gene sequence and submit it to netBLAST. Depending on the length and type of the query gene sequence, netBLAST could return more than ten matching sequences of any length. Since there is no way for the user to know a priori the number or the lengths of the resulting sequences, we implement navigation features within the inner-window such as zooming, panning, and scrolling to allow efficient visualization of the results. In addition, results can be moved to the working window for closer comparison, as described below.

To support random selection of partial or whole genomic sequence in conjunction with netBLAST, two different approaches are created. The user can either select a gene sequence from a complete genomic sequence using a sequence guideline, or from a partial sequence in an inner-window. Both methods can be accomplished by dragging the mouse over the displayed sequence or sequence guideline.

In addition to displaying the netBLAST results, the working window also shows the spatial correlation between similar sequences returned by netBLAST and their locations within the original genomic data. If the original genomic data is not present in our local database, our system will directly connect to NCBI database and download

it. The downloaded data will be integrated into the system through the steps such as gathering taxonomy information, finding ORFs⁴ data, generating a binary file based on collected data, and locating the file into one of the organism tree nodes. Thus GVis provides a powerful capability to build databases that are specific to users' needs and analyses and that contain the users own annotations. The database contains a history of the analysis for inspection by the user or sharing with others.

5.3 Knowledge Visualization

The knowledge visualization framework is applied to the initial version of GVis to increase the user's understanding of the represented knowledge and assist them in building tacit knowledge interactively. Throughout the rest of this thesis, we are going to use the same name "GVis" to represent the updated version of GVis. In the following sections, we explain how the knowledge visualization framework is applied and what features are supported.

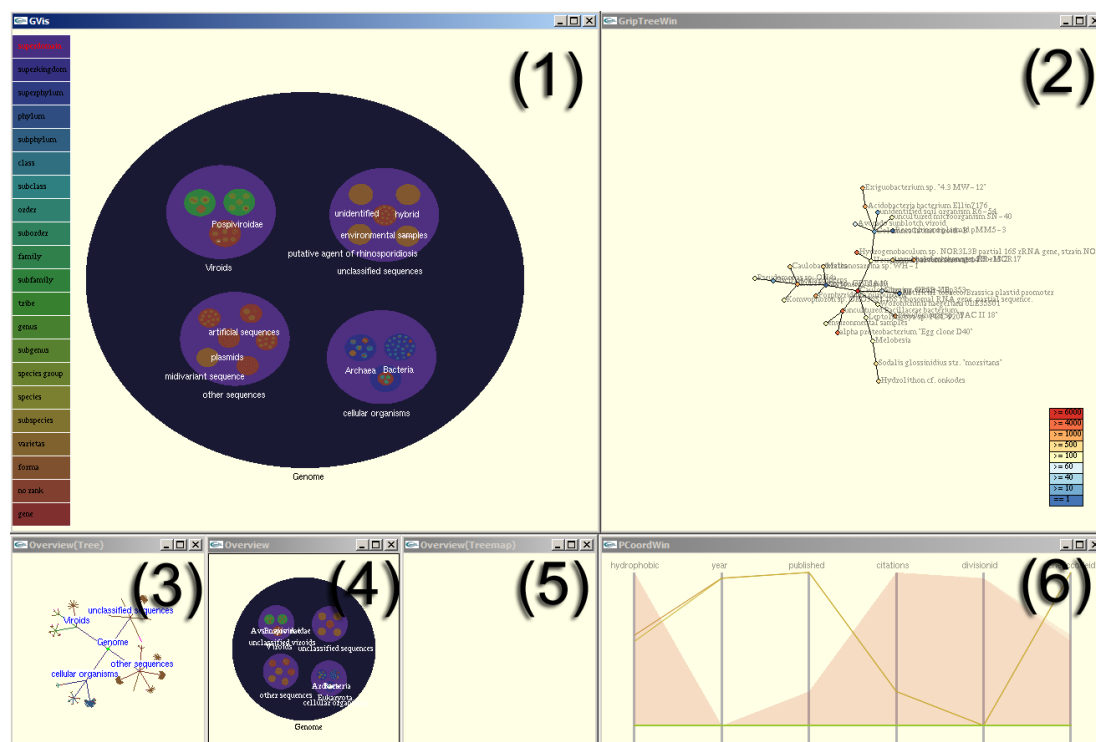
We applied several features that are highly related to the knowledge visualization framework. In particular, we displayed and interlinked a knowledge structure topology (nodes and links) to provide improved support for discovery, insight-building, and reasoning. We also provided interactions to build the user's own knowledge into the system, through new nodes or links, and to emphasize important knowledge, through selection and weight change in the visual displays. This provides both perceptual and cognitive benefits. Figure 58 shows the overview of GVis with the knowledge visualization framework applied. Similar to the initial version of GVis, it has a zoomable

⁴Open Reading Frame (ORF) is a portion of an organism's genome containing a sequence of bases (start- and end-points) that encode a protein.

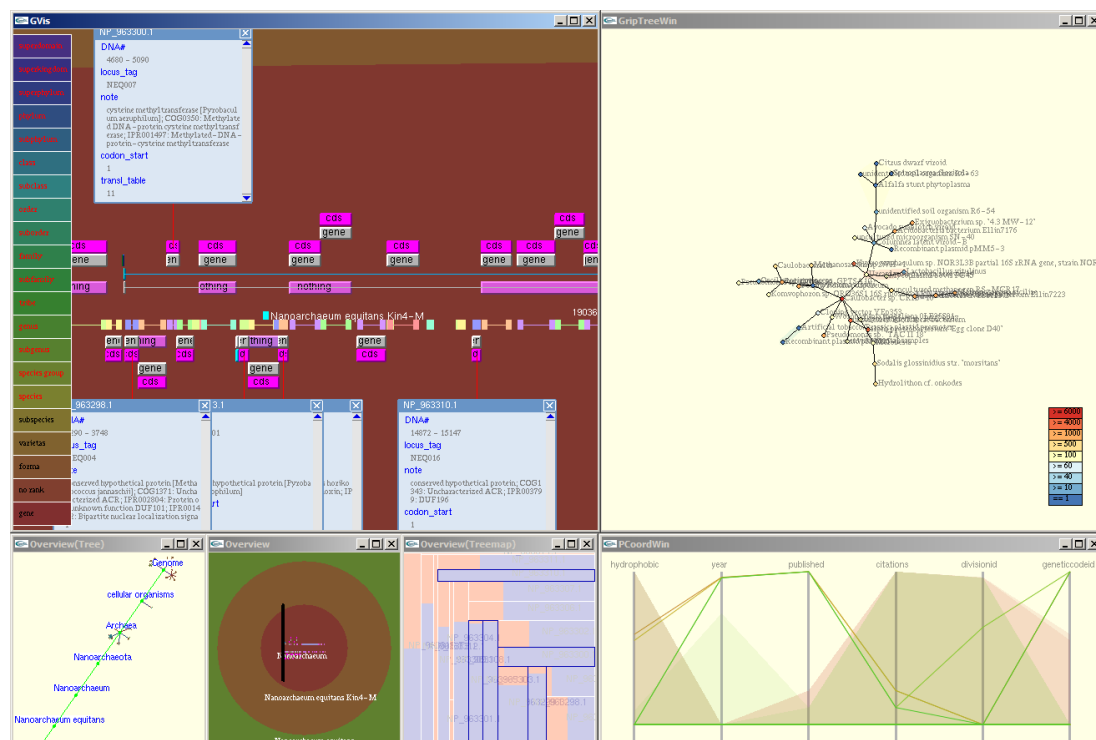
window (Figure 58(1)) and two overview windows (Figure 58(3) and (4)). Additional three windows are added to represent a knowledge structure topology and the cost and benefit of each item. In the knowledge structure window (Figure 58(2)), the relationship between genomic data are represented through which a user can interactively select data items to see their relationships closely. All the corresponding information about the knowledge structure topology displayed on screen is represented in parallel coordinates (Figure 58(6)). And the treemap view window (Figure 58(5)) shows the values (the cost and the benefit) of each data item where a treemap representation method is used to display the relative amount of cost (red-colored) and benefit (blue-colored). As shown, the selected items are highlighted in the treemap view window (see Figure 58(b)). A detailed explanation how to create the knowledge structure topology and how to handle knowledge (e.g. by applying the cost/ benefit analysis) in GVis are provided in the following sections.

5.3.1 Creating Knowledge Structure Topology

As we described in section 3.2, the knowledge visualization framework uses an ontological structure through using nodes, links, and classifications to show the relationships between concepts and entities. Specifically, in GVis, the taxonomy structure is represented as links in a hierarchical layout of organisms. Since the taxonomy does not show detailed relations among genomic data, it is important to provide a set of controlled terms (inferential structure) to describe gene and gene product attributes. A common approach to creating ontology structure is to use an inference engine. Often statistical approaches are used to create a structure that represents the relations



(a)



(b)

Figure 58: The overview of GVis with knowledge visualization framework applied. Users can interactively navigate from the initial view (a) to see the detailed representation within the genomic space (b).

among data [147, 57, 22]. In GVis, we use a statistical approach to build a graph topology (nodes and links) to represent the relations among all the organisms and the genomic data that augments the taxonomic layout. The data we use includes content from annotations created by researchers such as paper publications (published year and number of citations) or references to other organisms or to the taxonomy.

Based on more than 40,000 genomic entities, we applied three methods: PCA, Hierarchical clustering, and a force-directed drawing algorithm. Given an $n \times m$ data matrix, PCA uses the first eigenvectors of the $m \times m$ covariance matrix as the axes of the lower k -dimensional space [74]. The first two principal components are selected to represent each organism. An approximation approach, on-line SVD [14], is used to compute principal components in real-time. The data matrix can be computed in $O(mnr)$ time for $r \leq \sqrt{\min(m, n)}$. Within the principal components, a performance guaranteed hierarchical clustering technique proposed by Dasgupta and Long [41] is applied to build a hierarchical tree structure. It uses k -center algorithm and levels of granularity to approximately create a hierarchical clustering. It guarantees $\Omega(\log k)$ lower bound on the approximation with small values of k . If k is not defined, the complete linkage is computed in $O(n^2)$. However, representing the full graph topology in a limited screen space is not feasible. Also it is not possible to support interactive analysis of the graph topology [3]. Therefore, representative organisms are selected based on a distance-based approach. The force-directed drawing algorithm, GRIP [55], is used to efficiently organize each node (organism) to increase the user's awareness and understanding. Unnecessary or infeasible information automatically becomes inactive depending on the user's interest. This approach is useful to highlight

and represent important knowledge instead of representing all information without knowledge relations. It also shows that the creation of a knowledge structure takes less than 3 seconds for 40,000 genomic entities. This approach is useful to highlight and represent important knowledge instead of representing all information.

5.3.2 Labeling and Annotation

Labeling is important and highly connected to human cognitive processing. Controlled and well organized labels increase the ability of understanding visual contexts.

5.3.2.1 Applying the Cost/ Benefit Analysis

As we explained in section 5.2.1.2, GVis uses the annotation window to display a collection of genomic information (annotations about each gene). However, instead of simply representing all information, knowledge visualization focuses on emphasizing only important knowledge. To efficiently manage information and increase human's cognitive ability of understanding them, the cost/ benefit analysis (see section 3.2 for detail) is applied. In GVis, all available information is used to measure the benefit and the cost. The information are the size of gene sequences, nucleotide sequence identification numbers, a word or phrase describing about gene sequences, publication information, descriptions of genes and gene produces, regional information of biological significance related to gene sequences, etc. Among them, publications and the number of citations are important information to determine the benefit. Because of limited display space, the amount of information to be displayed on screen is regarded as an important factor to determine the cost. To measure the cost and the benefit, a greedy knapsack approach is used. If specific genomic data have a lot of

recent publications, higher number of citations, and the small amount of information, such genomic data as well as their corresponding annotations will be displayed on screen.

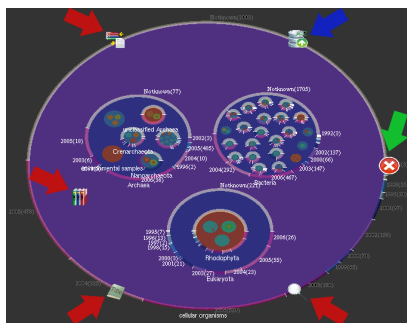
The benefit and the cost are computed every frame. Once a specific annotation window is displayed on screen, it is continuously shown on screen unless the user's viewing perspective has completely changed (see Figure 58(b)). If the visibility of information is frequently changed frame to frame, it diminishes the user's overall cognitive ability (hysteresis) of understanding it. Therefore, the benefit of maintaining displayed item on screen has a higher value than others. Since all genomic information are managed within each annotation window, the user can manually close the displayed annotation window. Once the annotation window is closed, the benefit and the cost will be recomputed.

The cost/ benefit analysis is also applied to control labels of organisms. Although many useful techniques have been proposed for rendering labels, most of the techniques require a pre-processing to determine what to display. However, with the cost/benefit analysis, all the labels of genomic data are managed on the fly when each label is displayed. To control each label, all labels are maintained to have the same value of benefit. And the cost is determined based on the length of each label and the distance to the closest label. It also guarantees a real-time computation to render the labels. With this approach, the user can navigate the scalable layout and see the labels without encountering cluttering.

5.3.2.2 Bringing Out the Information from the Bottom to the Top

In visualization, hierarchical clustering methods or taxonomy structures are often used to organize data in a limited display space. Since GVis uses a scalable layout based on a taxonomy tree to represent all genomic data, most raw data are positioned at the bottom layer. In such a layout, it is difficult for users to find useful information without navigating through the space. A circular way of information representation method is designed to remove this limitation. To initiate the action of showing information located at the bottom layer in front, Interactive-Icon tool is designed. Although the Interactive-Icon tool acts similarly to a traditional menu system, a recent paper by Green et al. [58] on the development of a human cognition model for visual analytics noted the importance of human interaction and flow of cognition in the model. They specifically pointed out that traditional menus cause considerable interruptions to an analyst's flow.

The Interactive-Icon tool uses six icons, each placed around the circumference of the organism's sphere (see Figure 59). Unlike the traditional menu system, the icons are dragged and dropped to the area of interest. For example, if the user wants to see the relevant information related to organism A, then an icon needs to be dropped in the region between the organism A and its subgroup organism(s). However, each icon does not represent data itself nor its symbolic meaning the way traditional icons do. Instead, it represents an operation to be performed. Four icons (see red arrows in Figure 59) can be dragged to active the pre-defined functionalities. Two additional icons (see green and blue arrows in Figure 59) used for sorting and for "canceling" the










Icon	Meaning	
	Published / Unpublished papers	
	Published years	
	Journal titles	
	Searching	
	Sorting (Possible)	
	Cancellation	

Figure 59: The five icons defined in Interactive-Icon tool.

In GVIs, the user’s focus is always changeable. Therefore, the Interactive-Icon tool adopts a focus-dependent representation method. The icons are automatically mapped on the circumference of interest. Also Interactive-Icon has an auto-hide feature that automatically hides the icons when navigating the larger genomic space in order to minimize visual clutter.

5.3.2.2.1 Representing Information

Whenever an option or icon is selected in the Interactive-Icon, the relevant information is represented around the organism circle's circumference.

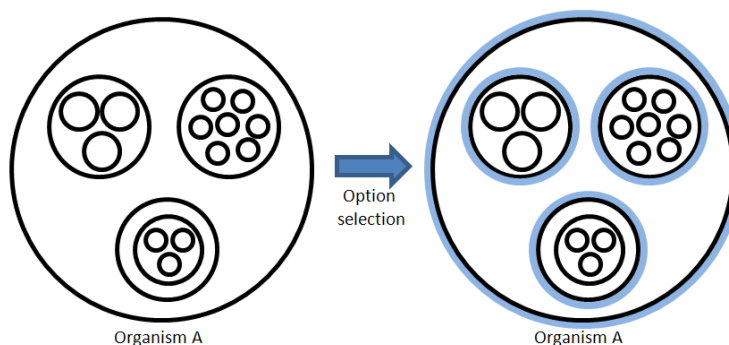


Figure 60: Search results are marked through use of color highlighting.

Figure 60 shows a simple example of how the information is represented in genomic space. Each circle represents an organism and the blue lines located around the organismic representation indicate the boundaries of each. The thickness of each blue line only changes during a Boolean search to highlight where these search results are located (see Figure 61(d)).

Figure 61 demonstrates how the each core functionality visualizes new information. All information is automatically aggregated before it is represented. The number of published and unpublished journal articles related to each organism is shown in Figure 61(a). Published papers are highlighted in yellow, unpublished in green. Figure 61(b) and (c) show more complex data, such as publication year and the title of the publishing journal. *Tone mapping* is used to delineate information. For example, Figure 61(d) highlights the number and location of search results by using a different line thickness in a high-contrast blue.

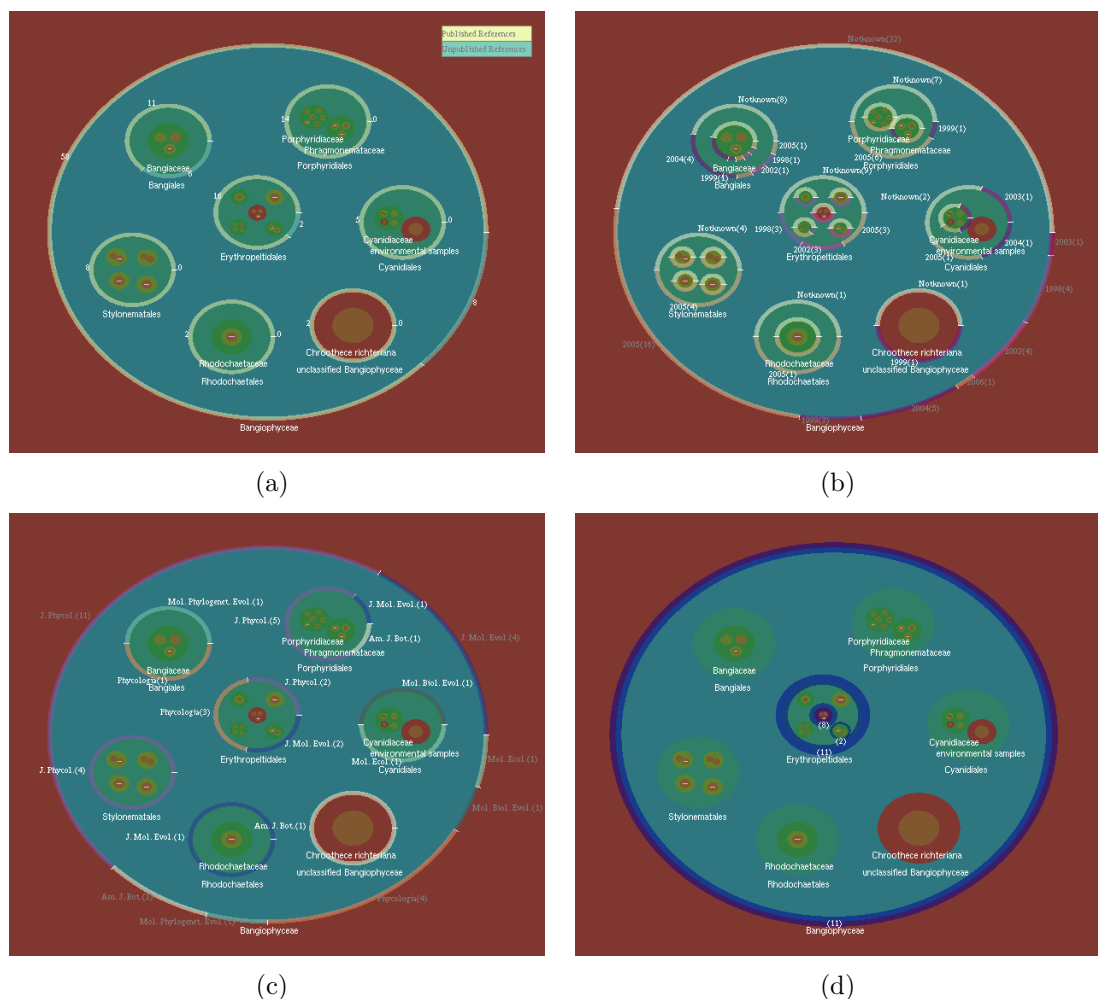


Figure 61: Four representations based on the selected core functionality: (a) number of published and unpublished articles, (b) year information about the publications, (c) titles of the publishing journals, and (d) result of a search. Three options (a - c) adopt a simple representation method to display information efficiently by representing the relevant information around the organism circle circumference. However, the searched result (d) shows the number of matched results by having different thickness of each line of each organism circle. The example (d) shows the result based on the input query “dales”, and indicate the matched result is found within the center organism group.

In genomic space, representing all information is not feasible since the scale of the visual space is limited. Because of this limitation, it always represents the information located within two levels (sub-organisms) down from the currently focused organism. Additionally, *labeling* is used to mitigate lexical clutter by continuously determining

the possibility of one word or phrase being overlapped with others through the scalable genomic space. By minimizing unnecessary clutter, the visualization design palliates attentional interference.

5.3.3 Interaction Supports

Interaction is an essential part of visualization and necessary to be supported to increase the users' ability of understanding the data or visual representation (e.g. static image) [157]. In knowledge visualization, similarly, interaction is regarded as a key component of assisting the user in understanding the presented knowledge and in building personalized knowledge. Within a scalable layout, the user can quickly browse the genomic data interactively. In GVis, a rich set of interactions are supported for user to discover and build knowledge interactively.

5.3.3.1 Knowledge Discovering

Representing data in a useful way is extremely important for users to understand the data. 2D or 3D representation methods are commonly used in visualization. However in 3D space, even if a good design approach has been adopted, it is still difficult for user to understand the data. This is because there is no one-to-one mapping for users to interact with a 2D user interface (such as a mouse - a pointing device) in 3D space. Since GVis is designed as a scalable layout (in 2.5D space), it is difficult for users to interact with the data. To minimize this limitation, GVis represent only important knowledge by measuring the values of knowledge. Also it provides a set of interactions such as zooming, selection, navigation, searching, etc.

The selection and navigation are the basic interactions commonly adopted visual-

ization applications. With the selection technique, users can perform selections on visual elements of interest. A set of selection techniques (individual item selection and group selection) are supported. Whenever the user select an item(s) in the genomic space, all the selected items are highlighted in other corresponding windows (*browsing & linking*). Since the genomic space in GVis is designed as a high-dimensional space, the navigation technique is useful for users to navigate the space freely. By navigating through the genomic space, users can easily find useful and important knowledge effectively. In addition, the Interactive-Icon tool (described in section 5.3.2) is an useful tool, with which users can interactively find important knowledge located at the bottom layer in the genomic space.

5.3.3.2 Knowledge Building

In knowledge visualization, building the user's own knowledge is as important as discovering knowledge. Instead of simply discovering knowledge in visualization, building knowledge is important to acquire the visualized knowledge as their personalized knowledge. To support this feature, direct manipulation technique is provided. After exploring the genomic space to discover knowledge, the user can manipulate the represented knowledge by performing linking, grouping, and annotating techniques. For instance, in the node-link diagram, users can change the representation by adding new links between nodes based on their background knowledge. The represented knowledge can be grouped together to form a new concept of knowledge by users. After knowledge building, the user can manage the representation by directly putting her personalized knowledge into the visualization.

5.3.4 Evaluation

To understand the effectiveness of the designed knowledge visualization application (GVis), we performed a study by comparing GVis with a well-known menu-driven web application (called MapViewer, Figure 62). Both applications use the same underlying dataset (GenBank). The MapViewer utilizes a multiple-row-based hierarchical representation and is manipulated through primary use of hyperlinks.

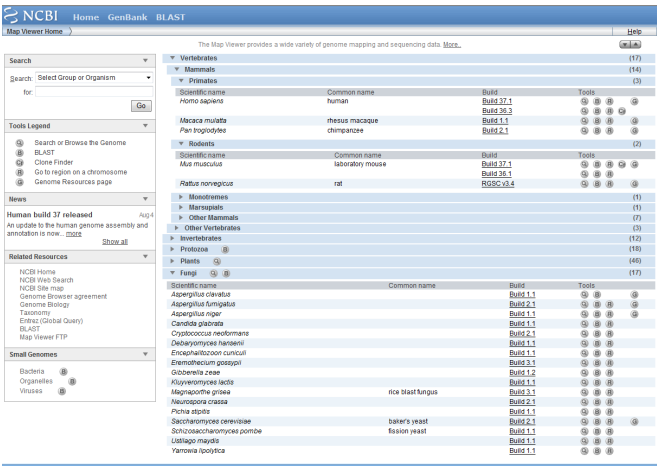


Figure 62: NCBI MapViewer

50 participants participated in the study. Among them only sixteen participants claimed that they had prior experience using data visualization applications. To minimize selection bias, we recruited college students who did not have expertise in biology. Each participant was requested to use the two applications. Therefore, twenty five participants used GVis first and the rest of the participants began with MapViewer.

5.3.4.1 Procedure

The study consists of four questionnaires including demographic, training, procedural, inferential, and post-study. All the questionnaires were managed using an on-line website. The procedural and inferential questionnaires were time-monitored. After completing the informed consent form, participants were requested to complete a demographic survey. During the training session, a brief demonstration how to use the evaluated tools was provided. Three training questions were used for the participants to become familiar with the applications. After the training session, the procedural and the inferential task questions were presented. The procedural questions asked the participants to identify a piece of information located somewhere within the presented informational hierarchy. As soon as they found the answer, the participants were requested to push a “Found It” button on the on-line website. Following the procedural questions, the participants was administered a series of tasks designed to test inferential performance. The list of questions for the applications is shown below.

[For GVis]

- Name another Pospiviroid that has the same number of total red groups and number of red groups with known genes as Columnnea Latent Viroid.
- Choose 2 ways that the Avsunviroidae are different from the Pospiviroidae.
 1. Only one contains viroids that attach vine plants.
 2. Only one has a known gene.
 3. Only one contains viroids that attack fruit plants.
 4. Only one has a known gene in every orange subgroup.

5. Only one has 5 orange subgroups.

- Which of the following orange subgroups is the most similar in hierarchical structure, characteristics, and labeling as the Citrus viroid IIIB?
 1. Citrus exocortis viroid (CEVdg-S)
 2. Apple scar skin viroid
 3. Coleus blumei viroid 3-RL
 4. Avocado sunblotch viroid V

[For MapViewer]

- Name another Betaproteobacteria organism that has the same number of organisms in its subgroups and same number of known genes in its subgroup as Verminephrobactereiseniae EF01-2.
- Choose 2 ways that Bordetella is different from Ralstonia.
 1. Only one contains organisms that cause respiratory ailments.
 2. Only one has more than 2 subgroups.
 3. Only one has a known gene in every subgroup.
 4. Only one has 4 known genes.
 5. Only one contains 4 or more subgroups with only 1 organism.
- Choose the organism in Betaproteobacteria that is the most similar in hierarchical structure, characteristics, and labeling to Ralstonia H16.
 1. Burkholderia ambifaria AMMD
 2. Palaromonas nephthalenivorans CJ2

3. *Bordetella bronchiseptica* RB50
4. *Neisseria gonorrhoeae* FA1090

Completion time and accuracy were recorded as outcome variables. After each participant completed the tasks in both applications, a post-study questionnaire was administered. Participants were asked to specify which interface they liked better and in which interface they were more comfortable working. They were also asked to freely respond to what they liked and disliked about each interface. Finally, they were asked to give each application a letter grade on a scale of 'A' to 'F'.

5.3.4.2 Evaluation Results

To complete procedural task questions, the participant spent about 136 ± 84 seconds with MapViewer and about 162 ± 111 seconds with GVis. A paired-sampled t-test between the total completion times in the procedural tasks across the interfaces achieved only borderline significance ($p = .057$). This nonsignificant trend is not entirely congruent with our expectation that we would not find a significant difference between the procedural completion times across interfaces. Although the participants spent less time to solve the procedural task questions, the trend was not strong enough to rule out random chance as a factor. This is because most participants are new to the visualization application.

However, for the inferential task, the participants spent less time with GVis. The participant spent about 451 ± 169 seconds with GVis and 922 ± 521 seconds with MapViewer. During the completion of the inferential tasks, participants answered more questions correctly while using the GVis ($M = 1.31, SD = .56$) interface

than while using the Mapviewer ($M = .77, SD = .59$). A paired-samples t-test between total inferential completion times for each interface was significant ($t(49) = -7.59, p < .01$), as were the total completion times (all 6 tasks) in both interfaces ($t(49) = 6.99, p < .01$). As shown in Table 7, all tasks completed in GVis were done more efficiently and effectively than those completed in the web application MapViewer.

Table 7: The number of participants who found the answer accurately for the inferential questions over two applications.

	Q1	Q2	Q3
GVis	40 (78%)	4(12%)	5 (10%)
MapViewer	2 (4%)	8 (16%)	2 (4%)

Correlations between total completion times were all significant: total procedural completion scores in both interfaces ($r(50) = .49, p < .01$), total inferential completion times in both interfaces ($r(50) = .61, p < .01$), and total completion times (all 6 tasks) in both interfaces ($r(50) = .63, p < .01$). These data demonstrate that participants who tended to take longer completing tasks in one interface also tended to take longer completing tasks in the other.

Overall, participants preferred interacting with the visualization to interacting with the web application. By asking the participants' preference (Figure 63), 36 (71%) of the participants gave the GVis an "A" or "B"; 18 (25%) gave an "A" or "B" to the MapViewer. Additionally, when asked, 33(66%) reported that they both preferred and were more comfortable in the visualization; 15 (30%) preferred and were more comfortable in the web application.

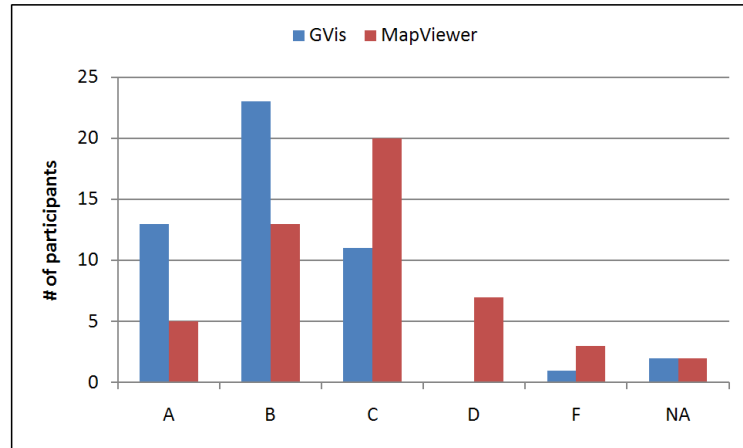


Figure 63: The participants' preference to use and to communicate with others.

Participants were also asked to freely respond to prompts about likes and dislikes in each interface. Common likes in GVis included the ease of use and use of color and graphical groupings; dislikes included not always knowing where to look for information. Common likes in MapViewer included its alphabetical and hierarchical organization; dislikes included difficult searches and the presentation of too much data.

5.3.4.3 Discussion

In this study, we demonstrate how the knowledge visualization is beneficial for learning and understanding knowledge. We did not find any statistically significant factor for solving the procedural task questions. However, for solving the inferential task questions, we noticed that the knowledge visualization is efficient to perform the tasks fast and accurately. Overall, we can conclude that when the tasks became more difficult, requiring the user to categorize, compare, and evaluate multiple choices at once, participants worked more quickly and made fewer errors while using the knowledge visualization.

CHAPTER 6: CONCLUSION AND FUTURE WORK

In this thesis, we propose a new research area “Knowledge Visualization.” Our work begins with understanding the existing definitions of knowledge. From the knowledge management literature, we found that knowledge can be classified as either tacit or explicit knowledge. Tacit knowledge refers to personalized knowledge, while explicit knowledge refers to formalized knowledge. We found that this distinction between tacit and explicit knowledge to be a good fit for a visualization model since knowledge visualization focuses on representing important knowledge instead of representing all data elements. In knowledge visualization, interaction is a key component to help users interactively understand the represented knowledge. Since knowledge visualization focuses on representing and interacting with knowledge in a visual space, it is necessary to clarify the definition of knowledge in visualization and to understand its relevant conversion processes. There are four knowledge conversion processes to consider: internalization, externalization, collaboration, and combination. To formalize the new concept of knowledge visualization, several projects have been conducted to understand each knowledge conversion process.

Based on the definitions of knowledge and the knowledge conversion processes in visualization, we designed a knowledge visualization framework. In this framework, a cost/benefit analysis method is defined to determine the value of knowledge as-

sociated with a dataset. According to its importance, the selected knowledge (data elements) is displayed to help the user understand complex data through visual analytical reasoning and discovery. Since this cost/benefit analysis is similar to a knapsack problem of maximizing benefit, a greedy knapsack approach is used. Overall, when applying the cost/benefit analysis, factors such as accuracy, value, hysteresis, and tidiness should be considered. Among them, two factors (hysteresis and tidiness) are especially important in terms of increasing a human's cognitive ability to understand the represented information in a limited display space. If the visibility of an object is frequently changed from frame to frame, it diminishes the users' cognitive ability to understand the object. Hysteresis is a "resistance to change" factor that prevents a scene from changing too much from frame to frame. Tidiness is used to control clutter (visual overlaps). It does this by using cell binning to concentrate on objects that are close to one another. To increase a user's ability to understand the visual representation, interaction is used to help the user to understand the presented knowledge and to build personalized knowledge from it.

We believe that the new concept of knowledge visualization promises a new paradigm in visualization. However, it is necessary and important to expand this concept and build it into a concrete model. Although the knowledge visualization application (GVis) has been proven to be useful for users to perform inferential task questions fast and accurately, it is important to perform an expert evaluation to examine the benefits and limitations of knowledge visualization.

REFERENCES

- [1] The American Heritage Medical Dictionary 2007.
- [2] Renaissance computing institute, 2009. <http://www.renci.org>.
- [3] ABELLO, J., VAN HAM, F., AND KRISHNAN, N. Ask-graphview: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 669–676.
- [4] ACKOFF, R. L. From data to wisdom. *Journal of Applied Systems Analysis* 16 (1989), 3–9.
- [5] AGRAFIOTIS, D. K., RASSOKHIN, D. N., AND LOBANOV, V. S. Multidimensional scaling and visualization of large molecular similarity tables. *Journal of Computational Chemistry* 22, 15 (2001), 488–500.
- [6] AHLBERG, C. Spotfire: An information exploration environment. *SIGMOD Record* 25, 4 (1996), 25–29.
- [7] AHLBERG, C., WILLIAMSON, C., AND SHNEIDERMAN, B. Dynamic queries for information exploration: an implementation and evaluation. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1992), ACM, pp. 619–626.
- [8] ALTSCHUL, S., GISH, W., MILLER, W., MYERS, E., AND LIPMAN, D. Basic local alignment search tool. *Journal of Molecular Biology* 215, 3 (1990), 403–410.
- [9] ARIADNE GENOMICS. PathwayStudioTM. <http://www.ariadnegenomics.com>.
- [10] ASUNCION, A., AND NEWMAN, D. UCI machine learning repository, 2007.
- [11] BEDERSON, B. B., AND HOLLAN, J. D. Pad++: a zooming graphical interface for exploring alternate interface physics. In *UIST '94: Proceedings of the 7th annual ACM symposium on User interface software and technology* (New York, NY, USA, 1994), ACM, pp. 17–26.
- [12] BINNS, J., DICARLO, J., INSLEY, J. A., LEGGETT, T., LUENINGHOENER, C., NAVARRO, J.-P., AND PAPKA, M. E. Enabling community access to teragrid visualization resources: Research articles. *Concurr. Comput. : Pract. Exper.* 19, 6 (2007), 783–794.
- [13] BONDY, J., AND MURTY, U. *Graph Theory With Applications*. Elsevier Science Ltd, 1976.

- [14] BRAND, M. Fast online svd revisions for lightweight recommender systems. In *SDM 2003: Proceedings of the SIAM International Conference on Data Mining* (2003), pp. 83–91.
- [15] BRAND, M. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications* 415, 1 (2006), 20–30.
- [16] BREINHOLT, G., AND SCHIERZ, C. Algorithm 781: generating hilbert’s space-filling curve by recursion. *ACM Transactions on Mathematical Software (TOMS)* 24, 2 (1998), 184–189.
- [17] BROWN, J., MCGREGOR, A., AND BRAUN, H. Network performance visualization: insight through animation. In *PAM 2000: Proceedings of the Passive and Active Measurement Workshop* (2000), pp. 33–41.
- [18] BUJA, A., McDONALD, J. A., MICHALAK, J., AND STUETZLE, W. Interactive data visualization using focusing and linking. In *VIS ’91: Proceedings of the 2nd conference on Visualization ’91* (Los Alamitos, CA, USA, 1991), IEEE Computer Society Press, pp. 156–163.
- [19] BURKHARD, R. Learning from architects: the difference between knowledge visualization and information visualization. In *IV 2004: Proceedings of the Information Visualisation, Eighth International Conference* (July 2004), pp. 519–524.
- [20] BUTKIEWICZ, T., JEONG, D. H., RIBARSKY, W., AND CHANG, R. Hierarchical multi-touch selection techniques for collaborative geospatial analysis. In *Proceedings of the SPIE Defense, Security, and Sensing 2009* (2009).
- [21] CAHN, S. M. *Classics of Western Philosophy*. Hackett Publishing Company, 1977.
- [22] CAI, Z., MAO, X., LI, S., AND WEI, L. Genome comparison using gene ontology (go) with statistical testing. *BMC Bioinformatics* 7, 374 (2006).
- [23] CALLAHAN, E., AND KOENEMANN, J. A comparative usability evaluation of user interfaces for online product catalog. In *EC ’00: Proceedings of the 2nd ACM conference on Electronic commerce* (New York, NY, USA, 2000), ACM, pp. 197–206.
- [24] CARLBOM, I., HSU, W. M., KLINKER, G., SZELISKI, R., WATERS, K., DOYLE, M., GETTYS, J., HARRIS, K. M., LEVERGOOD, T. M., PALMER, R., PALMER, L., PICART, M., TERZOPOULOS, D., TONNESEN, D., VANNIER, M., AND WALLACE, G. Modeling and analysis of empirical data in collaborative environments. *Commun. ACM* 35, 6 (1992), 74–84.
- [25] CHANG, R., GHONIEM, M., KOSARA, R., RIBARSKY, W., YANG, J., SUMA, E., ZIEMKIEWICZ, C., KERN, D., AND SUDJANTO, A. Wirevis: Visualization

- of categorical, time-varying data from financial transactions. *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on* (30 2007-Nov. 1 2007), 155–162.
- [26] CHANG, R., ZIEMKIEWICZ, C., GREEN, T. M., AND RIBARSKY, W. Defining insight for visual analytics. *IEEE Computer Graphics and Applications* 29, 2 (2009), 14–17.
 - [27] CHEN, M., EBERT, D., HAGEN, H., LARAMEE, R., VAN LIERE, R., MA, K.-L., RIBARSKY, W., SCHEUERMANN, G., AND SILVER, D. Data, information, and knowledge in visualization. *Computer Graphics and Applications, IEEE* 29, 1 (Jan.-Feb. 2009), 12–19.
 - [28] CHIN, JR., G., KUCHAR, O. A., AND WOLF, K. E. Exploring the analytical processes of intelligence analysts. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems* (New York, NY, USA, 2009), ACM, pp. 11–20.
 - [29] CHO, R., CAMPBELL, M., WINZELER, E., STEINMETZ, L., CONWAY, A., WODICKA, L., WOLFSBERG, T., GABRIELIAN, A., LANDSMAN, D., LOCKHART, D., AND DAVIS, R. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell* 2, 1 (2000), 65–73.
 - [30] CHUAH, M., ROTH, S., MATTIS, J., AND KOLOJEJCHICK, J. Sdm: malleable information graphics. In *Proceedings of Information Visualization '95 Symposium* (Oct. 1995), pp. 36–42.
 - [31] COLEMAN, J., GOETTSCH, A., SAVCHENKO, A., KOLLMANN, H., WANG, K., KLEMENT, E., AND BONO, P. TeleinvivoTM: Towards collaborative volume visualization environments. *Computer & Graphics* 20, 6 (1996), 801–811.
 - [32] COMBS, T. T. A., AND BEDERSON, B. B. Does zooming improve image browsing? In *DL '99: Proceedings of the fourth ACM conference on Digital libraries* (New York, NY, USA, 1999), ACM, pp. 130–137.
 - [33] COWLEY, P., HAACK, J., LITTLEFIELD, R., AND HAMPSON, E. Glass box: capturing, archiving, and retrieving workstation activities. In *CARPE '06: Proceedings of the 3rd ACM workshop on Continuous archival and retrieval of personal experiences* (2006), pp. 13–18.
 - [34] COWLEY, P., NOWELL, L., AND SCHOLTZ, J. Glass box: An instrumented infrastructure for supporting human interaction with information. In *HICSS '05: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005*. (January 2005), p. 296c.
 - [35] CRAIG, P., KENNEDY, J., AND CUMMING, A. Towards visualising temporal features in large scale microarray time-series data. In *Proceedings of Sixth International Conference on Information Visualisation, 2002* (2002), pp. 427–433.

- [36] CRAIG, P., KENNEDY, J., AND CUMMING, A. Coordinated parallel views for the exploratory analysis of microarray time-course data. In *CMV '05: Proceedings of the Coordinated and Multiple Views in Exploratory Visualization* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 3–14.
- [37] CROWLEY, B. Tacit knowledge, tacit ignorance, and the future of academic librarianship. *College & Research Libraries* 62 (2001), 565–584.
- [38] DAHLQUIST, K., SALOMONIS, N., VRANIZAN, K., LAWLOR, S., AND CONKLIN, B. Genmapp, a new tool for viewing and analyzing microarray data on biological pathways. *Nature Genetics* 31, 1 (2002), 19–20.
- [39] DARVISH, A., HAKIMZADEH, R., AND NAJARIAN, K. Discovering dynamic regulatory pathway by applying an auto regressive model to time series dna microarray data. In *IEMBS '04: Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2004* (Sept. 2004), vol. 2, pp. 2941–2944.
- [40] DARVISH, A., NAJARIAN, K., JEONG, D., AND RIBARSKY, W. System identification and nonlinear factor analysis for discovery and visualization of dynamic gene regulatory pathways. In *CIBCB '05: Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, 2005* (Nov. 2005), pp. 1–6.
- [41] DASGUPTA, S., AND LONG, P. M. Performance guarantees for hierarchical clustering. *J. Comput. Syst. Sci.* 70, 4 (2005), 555–569.
- [42] DAVENPORT, T. H., AND PRUSAK, L. *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, 1998.
- [43] DE HOON, M., IMOTO, S., AND MIYANO, S. Statistical analysis of a small set of time-ordered gene expression data using linear splines. *Bioinformatics* 8, 11.
- [44] DE WAELE, S., AND BROERSEN, P. Order selection for vector autoregressive models. *Signal Processing, IEEE Transactions on* 51, 2 (Feb 2003), 427–433.
- [45] DILLON, K. M., AND TALBOT, P. J. Knowledge visualization: Redesigning the human-computer interface. *Technology Review Journal* (2007), 37–55.
- [46] DRUCKER, P. F. *Post-Capitalist Society*. Harper Business, 1993.
- [47] EISEN, M., SPELLMAN, P., BROWN, P., AND BOTSTEIN, D. Cluster analysis and display of genome-wide expression patterns. *PNAS* 95, 25 (1998), 14863–14868.
- [48] EISENSTEIN, M. Microarrays: Quality control. *Nature* 442 (2006), 1067–1070.

- [49] EKANAYAKE, J., PALICKARA, S., AND FOX, G. A collaborative framework for scientific data analysis and visualization. In *CTS 2008: Proceedings of the International Symposium on Collaborative Technologies and Systems, 2008* (May 2008), pp. 339–346.
- [50] FEKETE, J.-D., AND PLAISANT, C. Excentric labeling: dynamic neighborhood labeling for data visualization. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1999), ACM, pp. 512–519.
- [51] FINK, G. A., NORTH, C. L., ENDERT, A., AND ROSE, S. Visualizing cyber security: Usable workspaces. In *VizSec '09: Proceedings of the 6th International Workshop on Visualization for Cyber Security* (2009).
- [52] FRIEDMAN, N., LINIAL, M. AND NACHMAN, I., AND PE'ER, D. Using bayesian network to analyze expression data. *Journal of Computational Biology* 7 (2000), 601–620.
- [53] FUNKHOUSER, T. A., AND SÉQUIN, C. H. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1993), ACM, pp. 247–254.
- [54] FURNAS, G. W., AND BEDERSON, B. B. Space-scale diagrams: understanding multiscale interfaces. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1995), ACM Press/Addison-Wesley Publishing Co., pp. 234–241.
- [55] GAJER, P., AND KOBOUROV, S. G. Grip: Graph drawing with intelligent placement. In *Proceedings of the Graph Drawing 2000* (2000), Springer-Verlag, pp. 222–228.
- [56] GARG, S., NAM, J., RAMAKRISHNAN, I., AND MUELLER, K. Model-driven visual analytics. In *VAST '08: Proceedings of the IEEE Symposium on Visual Analytics Science and Technology, 2008*. (Oct. 2008), pp. 19–26.
- [57] GÓMEZ-PÉREZ, A., AND MANZANO-MACHO, D. An overview of methods and tools for ontology learning from texts. *Knowl. Eng. Rev.* 19, 3 (2004), 187–212.
- [58] GREEN, T., RIBARSKY, W., AND FISHER, B. Visual analytics for complex concepts using a human cognition model. In *VAST '08: Proceedings of the IEEE Symposium on Visual Analytics Science and Technology, 2008* (Oct. 2008), pp. 91–98.
- [59] GREITZER, F. Methodology, metrics and measures for testing and evaluation of intelligence analysis tools. PNWD-3550, Battelle-Pacific Northwest Division, Richland, WA, 2005.

- [60] GRIMSTEAD, I. J., WALKER, D. W., AND AVIS, N. J. Collaborative visualization: A review and taxonomy. *Distributed Simulation and Real-Time Applications, IEEE International Symposium on 0* (2005), 61–69.
- [61] GRUBER, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition* 5, 2 (1993), 199–220.
- [62] HEER, J., AND AGRAWALA, M. Design considerations for collaborative visual analytics. *Information Visualization* 7, 1 (2008), 49–62.
- [63] HEER, J., MACKINLAY, J. D., STOLTE, C., AND AGRAWALA, M. Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1189–1196.
- [64] HIBBS, M. A., DIRKSEN, N. C., LI, K., AND TROYANSKAYA, O. G. Visualization methods for statistical analysis of microarray clusters. *BMC Bioinformatics* 6, 115 (2005).
- [65] HONG, J., JEONG, D. H., SHAW, C. D., RIBARSKY, W., BORODOVSKY, M., AND SONG, C. Gvis: A scalable visualization framework for genomic data. In *EUROVIS 2005: Proceedings of the Eurographics / IEEE VGTC Symposium on Visualization 2005* (2005), pp. 191–198.
- [66] IBM. Many eyes. <http://manyeyes.alphaworks.ibm.com/manyeyes>.
- [67] ISENBERG, P., BEZERIANOS, A., HENRY, N., CARPENDALE, S., AND FEKETE, J.-D. Coconuttrix: Collaborative retrofitting for information visualization. *IEEE Computer Graphics and Applications* 29, 5 (2009), 44–57.
- [68] ISENBERG, P., AND CARPENDALE, S. Interactive tree comparison for co-located collaborative information visualization. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (2007), 1232–1239.
- [69] ISENBERG, P., AND FISHER, D. Collaborative brushing and linking for co-located visual analytics of document collections. *Computer Graphics Forum (EuroVis 2009)* 28, 3 (2009), 1031–1038.
- [70] JANKUN-KELLY, T., MA, K.-L., AND GERTZ, M. A model and framework for visualization exploration. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (March-April 2007), 357–369.
- [71] JEONG, D. H., CHANG, R., AND RIBARSKY, W. An alternative definition and model for knowledge visualization. IEEE Visualization 2008 Workshop on Knowledge Assisted Visualization, 2008.
- [72] JEONG, D. H., ZIEMKIEWICZ, C., FISHER, B., RIBARSKY, W., AND CHANG, R. iPCA: An interactive system for pca-based visual analytics. *Computer Graphics Forum (EuroVis 2009)* 28, 3 (2009), 767–774.

- [73] JOHNSON, G. Collaborative visualization 101. *ACM SIGGRAPH - Computer Graphics* 32, 2 (1999), 8–11.
- [74] JOLLIFFE, I. T. *Principal Component Analysis*, second ed. Springer, 2002.
- [75] JUDELMAN, G. B. Knowledge visualization: Problems and principles for mapping the knowledge space. Master's thesis, University of Lubeck, Germany, 2004.
- [76] JURISICA, I., AND WIGLE, D. *Knowledge Discovery in Proteomics*, second ed. CRC Press, 2005.
- [77] KEIM, D. A. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics* 6, 1 (2000), 59–78.
- [78] KEIM, D. A., KRIEGEL, H.-P., AND SEIDL, T. Visual feedback in querying large databases. In *VIS '93: Proceedings of the 4th conference on Visualization '93* (Washington, DC, USA, 1993), IEEE Computer Society, pp. 158–165.
- [79] KELLERER, H., PFERSCHY, U., AND PISINGER, D. *Knapsack Problems*, first ed. Springer, 2004.
- [80] KOIKE, A., AND TAKAGI, T. Knowledge discovery based on an implicit and explicit conceptual network. *Journal of the American Society for Information Science and Technology* 58 (2006), 51–65.
- [81] LIBRARY OF CONGRESS. Library of congress. <http://www.loc.gov/about/generalinfo.html>.
- [82] LIU, X., MININ, V., HUANG, Y., SELIGSON, D., AND HORVATH, S. Statistical methods for analyzing tissue microarray data. *Journal of biopharmaceutical statistics* 14, 3 (2004), 671–85.
- [83] LUO, H., FAN, J., SATOH, S., AND RIBARSKY, W. Large scale news video database browsing and retrieval via information visualization. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing* (New York, NY, USA, 2007), ACM, pp. 1086–1087.
- [84] MA, K.-L. Creating a collaborative space to share data, visualization, and knowledge. *ACM SIGGRAPH - Computer Graphics* 41, 4 (2007), 1–4.
- [85] MACHLUP, F. *Knowledge, its creation, distribution, and economic significance*. Princeton University Press, 1982.
- [86] MACKINLAY, J., HANRAHAN, P., AND STOLTE, C. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (2007), 1137–1144.

- [87] MARCHESE, F. T., AND BRAJKOVSKA, N. Fostering asynchronous collaborative visualization. In *IV '07: Proceedings of the 11th International Conference Information Visualization* (Washington, DC, USA, 2007), IEEE Computer Society, pp. 185–190.
- [88] MARK, G., AND KOBZA, A. The effects of collaboration and system transparency on cive usage: An empirical study and model. *Presence* 14 (2005), 60–80.
- [89] MCGUFFIN, M. J., TANCAU, L., AND BALAKRISHNAN, R. Using deformations for browsing volumetric data. In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)* (Washington, DC, USA, 2003), IEEE Computer Society, p. 53.
- [90] MCNEILL, J., STUESSY, T. F., TURLAND, N. J., AND HORANDL, E. Xvii international botanical congress: preliminary mail vote and report of congress action on nomenclature proposals. *Taxon* 54, 4 (2005), 8.
- [91] MEISSNER, W. W. *Internalization in Psychoanalysis*. New York: International Universities Press, 1981.
- [92] MOSER, R., REVERTER, A., KERR, C., BEH, K., AND LEHNERT, S. A mixed-model approach for the analysis of cDNA microarray gene expression data from extreme-performing pigs after infection with actinobacillus pleuropneumoniae. *Journal of Animal Science* 82 (2004), 1261–1271.
- [93] MÜLLER, W., AND ALEXA, M. Visual component analysis. In *VisSym 2004: Proceedings of Joint IEEE/EG Symposium on Visualization* (2004), Eurographics Association, pp. 129–136.
- [94] MÜLLER, W., NOCKE, T., AND SCHUMANN, H. Enhancing the visualization process with principal component analysis to support the exploration of trends. In *APVis '06: Proceedings of the Asia Pacific Symposium on Information on Information Visualisation* (2006), Australian Computer Society, Inc., pp. 121–130.
- [95] NAKAHARA, H., NISHIMURA, S., INOUE, M., HORI, G., AND AMARI, S. Gene interaction in dna microarray data is decomposed by information geometric measure. *Bioinformatics* 19, 9 (2003), 1124–1131.
- [96] NATIONAL WEATHER SERVICE. National doppler radar sites. <http://radar.weather.gov>.
- [97] NONAKA, I., AND TAKEUCHI, H. *The Knowledge Creating Company*. Oxford University Press, 1995.
- [98] PACIFIC NORTHWEST NATIONAL LABORATORY (PNNL). Inspire. <http://in-spire.pnl.gov>, 2008.

- [99] PERLIN, K., AND FOX, D. Pad: an alternative approach to the computer interface. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1993), ACM, pp. 57–64.
- [100] PERRIN, B., RALAIVOLA, L., MAZURIE, A., BOTTANI, S., MALLET, J., AND D'ALCHE-BUC, F. Gene networks inference using dynamic bayesian networks. *Bioinformatics* 19, Suppl.2 (2003), II138–II148.
- [101] PETER LYMAN AND HAL R. VARIAN . How much information 2003? <http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/>.
- [102] PETERSON, B. Dynamic visualization of microarray time series. <http://www.terastat.com>, 2005.
- [103] PIKE, W., MAY, R., AND TURNER, A. Supporting knowledge transfer through decomposable reasoning artifacts. In *HICSS '07: Proceedings of the 40th Annual Hawaii International Conference on System Sciences, 2007*. (January 2007), p. 204c.
- [104] PIKE, W. A., STASKO, J., CHANG, R., AND O'CONNELL, T. A. The science of interaction. *Information Visualization* 8 (2009), 263–74.
- [105] PIRINGER, H., TOMINSKI, C., MUIGG, P., AND BERGER, W. A multi-threading architecture to support interactive visual exploration. *IEEE Transactions on Visualization and Computer Graphics* 15 (2009), 1113–1120.
- [106] PIROLI, P., CARD, S. K., AND VAN DER WEGE, M. M. Visual information foraging in a focus + context visualization. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2001), ACM, pp. 506–513.
- [107] POLANYI, M. *Tacit Dimension*. Peter Smith Publisher, 1983.
- [108] POPPER, K. R. *Objective Knowledge: An Evolutionary Approach*. Oxford University Press, 1972.
- [109] PRETORIUS, A. J., AND VAN WIJK, J. J. Bridging the semantic gap: Visualizing transition graphs with user-defined diagrams. *IEEE Computer Graphics and Applications* 27, 5 (2007), 58–66.
- [110] REICHERT, J., JABS, A., P., S., AND J., S. The imb jena image library of biological macromolecules. *Nucleic Acids Research* 28, 1 (2000), 246–249.
- [111] RIBARSKY, W., FAUST, N., WARTELL, Z., SHAW, C., AND JANG, J. Visual query of time-dependent 3d weather in a global geospatial environment. In *Mining Spatio-temporal Information Systems*, R. Ladner, K. Shaw, and M. Abdelguerfi, Eds. Springer-Verlag, 2002, ch. 5, pp. 83–104.

- [112] ROBERTSON, G. G., MACKINLAY, J. D., AND CARD, S. K. Information visualization using 3d interactive animation. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1991), ACM, pp. 461–462.
- [113] ROUCHKA, E. C., MAZZARELLA, R., AND STATES, D. J. Computational detection of cpg islands in dna. <http://www.yeastgenome.org>.
- [114] SALES-PARDO, M., GUIMERÀ, R., MOREIRA, A. A., WIDOM, J., AND AMARAL, L. A. N. Mesoscopic modeling for nucleic acid chain dynamics. *Physical Review E* 71, 5 (May 2005), 051902.
- [115] SALVADOR, T., SCHOLTZ, J., AND LARSON, J. The denver model for groupware design. *SIGCHI Bull.* 28, 1 (1996), 52–58.
- [116] SARAIYA, P., NORTH, C., AND DUCA, K. Visualizing biological pathways: requirements analysis, systems evaluation and research agenda. *Information Visualization* 4, 3 (2005), 191–205.
- [117] SARAIYA, P., NORTH, C., LAM, V., AND DUCA, K. A. An insight-based longitudinal study of visual analytics. *IEEE Transactions on Visualization and Computer Graphics* 12, 6 (2006), 1511–1522.
- [118] SAS INSTITUTE, INC. SAS/INSIGHT. <http://sas.com/technologies/analytics/statistics/insight>.
- [119] SCHULZE-WOLLGAST, P., TOMINSKI, C., AND SCHUMANN, H. Enhancing visual exploration by appropriate color coding. In *WSCG'05: Proceedings of International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* (2005), pp. 203–210.
- [120] SCOTT, S. D., SHEELAGH, M., CARPENDALE, T., AND INKPEN, K. M. Territoriality in collaborative tabletop workspaces. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work* (New York, NY, USA, 2004), ACM, pp. 294–303.
- [121] SEO, J., AND SHNEIDERMAN, B. Knowledge discovery in high-dimensional data: Case studies and a user survey for the rank-by-feature framework. *IEEE Transactions on Visualization and Computer Graphics* 12, 3 (2006), 311–322.
- [122] SHLENS, J. A tutorial on principal component analysis. <http://www.sn1.salk.edu/~shlens/notes.html>, 2005.
- [123] SHRINIVASAN, Y. B., AND VAN WIJK, J. J. Supporting the analytical reasoning process in information visualization. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2008), ACM, pp. 1237–1246.
- [124] SILICON GENETICS. GeneSpringTM. <http://www.silicongenetics.com>.

- [125] SILVESCU, A., AND HONAVAR, V. Temporal boolean network models of genetic networks and their inference from gene expression time series. *Complex Systems* 13, 1 (2001), 54–70.
- [126] SNIEZYNSKI, B., SZYMACHA, R., AND MICHALSKI, R. S. Knowledge visualization using optimized general logic diagrams. In *IIPWM 05: Proceedings of the Intelligent Information Processing and Web Mining Conference* (2005), pp. 137–146.
- [127] SPEED, T. P. *Statistical analysis of gene expression microarray data*, 1 ed. Interdisciplinary statistics. Chapman & Hall/CRC, March 2003.
- [128] SYED, A., AND SHAH, A. Data, information, knowledge, wisdom: A doubly linked chain? In *IKE 2006: Proceedings of the International Conference on Information and Knowledge Engineering* (2006), pp. 270–278.
- [129] SYMEONIDIS, A., AND TOLLIS, I. G. Visualization of biological information with circular drawings. In *ISBMDA '04: Proceedings of the Biological and Medical Data Analysis* (2004), pp. 468–478.
- [130] TANG, A., TORY, M., PO, B., NEUMANN, P., AND CARPENDALE, S. Collaborative coupling over tabletop displays. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems* (New York, NY, USA, 2006), ACM, pp. 1181–1190.
- [131] THE BOARD OF TRUSTEES, LELAND STANFORD JUNIOR UNIVERSITY. Saccharomyces genome database. <http://www.yeastgenome.org>.
- [132] THE GGOBI FOUNDATION, INC. Ggobi. <http://www.ggobi.org>.
- [133] THE MATHWORKS, INC. Matlab. <http://www.mathworks.com/products/matlab>.
- [134] THE NATIONAL HUMAN GENOME RESEARCH INSTITUTE (NHGRI). The gene ontology. <http://www.geneontology.org>.
- [135] THE UNIVERSITY OF OKLAHOMA. Cooperative institute for mesoscale meteorological studies (cimms). <http://www.cimms.ou.edu>.
- [136] THOMAS, J., AND COOK, K. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005.
- [137] THOMAS, J. J., AND COOK, K. A. A visual analytics agenda. *IEEE Computer Graphics and Applications* 26, 1 (2006), 10–13.
- [138] TOMINSKI, C., SCHULZE-WOLLGAST, P., AND SCHUMANN, H. 3d information visualization for time dependent data on maps. In *IV '05: Proceedings of the Ninth International Conference on Information Visualisation* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 175–181.

- [139] TOYODA, T., AND KONAGAYA, A. Knowledgeeditor: a new tool for interactive modeling and analyzing biological pathways based on microarray data. *Bioinformatics* 19, 3 (2003), 433–434.
- [140] TSAI, W.-T., VISHNUVAJJALA, R., AND ZHANG, D. Verification and validation of knowledge-based systems. *Knowledge and Data Engineering, IEEE Transactions on* 11, 1 (Jan/Feb 1999), 202–212.
- [141] TSE, E., HISTON, J., SCOTT, S. D., AND GREENBERG, S. Avoiding interference: how people use spatial separation and partitioning in sdg workspaces. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work* (New York, NY, USA, 2004), ACM, pp. 252–261.
- [142] TUDDENHAM, P., AND ROBINSON, P. Territorial coordination and workspace awareness in remote tabletop collaboration. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems* (New York, NY, USA, 2009), ACM, pp. 2139–2148.
- [143] VAN WEZEL, M. C., AND KOSTERS, W. A. Nonmetric multidimensional scaling: Neural networks versus traditional techniques. *Intelligent Data Analysis* 8, 6 (2004), 601–613.
- [144] VAN WIJK, J. J. The value of visualization. In *VIS 05: Proceedings of the IEEE Symposium on Visual Analytics Science And Technology, 2005* (2006), pp. 79–86.
- [145] WALL, M. E., RECHTSTEINER, A., AND ROCHA, L. M. Singular value decomposition and principal component analysis. In *A Practical Approach to Microarray Data Analysis* (D.P. Berrar, W. Dubitzky, M. Granzow, eds.) Kluwer: Norwell, MA, pp. 91–109., 2003.
- [146] WALLACE, D. P. *Knowledge management: historical and cross-disciplinary themes*, second ed. Libraries Unlimited, 2007.
- [147] WANG, B. B., MCKAY, R. I. B., ABBASS, H. A., AND BARLOW, M. A comparative study for domain ontology guided feature extraction. In *ACSC '03: Proceedings of the 26th Australasian computer science conference* (Darlinghurst, Australia, 2003), Australian Computer Society, Inc., pp. 69–78.
- [148] WENWEN ET AL., D. Recovering reasoning processes from user interactions. *IEEE Computer Graphics and Applications* 29, 3 (2009), 52–61.
- [149] WHEELER, D. L., CHURCH, D. M., FEDERHEN, S., LASH, A. E., MADDEN, T. L., PONTIUS, J. U., SCHULER, G. D., SCHRIML, L. M., SEQUEIRA, E., TATUSOVA, T. A., AND WAGNER, L. Database resources of the National Center for Biotechnology. *Nucleic Acids Research* 31, 1362–4962 (Electronic) (2003), 28–33.

- [150] WISE, J., THOMAS, J., PENNOCK, K., LANTRIP, D., POTTIER, M., SCHUR, A., AND CROW, V. Visualizing the non-visual: spatial analysis and interaction with information from text documents. *IEEE Symposium on Information Visualization 0* (1995), 51.
- [151] WOLFSBERG, T., GABRIELIAN, A., CAMPBELL, M., CHO, R., SPOUGE, J., AND LANDSMAN, D. Candidate regulatory sequence elements for cell cycle-dependent transcription in *saccharomyces cerevisiae*. *Genome Research* 9, 8 (1999), 775–792.
- [152] WONG, P. C., FOOTE, H., MACKEY, P., PERRINE, K., AND JR., G. C. Generating graphs for visual analytics through interactive sketching. *IEEE Transactions on Visualization and Computer Graphics* 12, 6 (2006), 1386–1398.
- [153] WONG, P. C., WONG, K. K., FOOTE, H., AND THOMAS, J. Global visualization and alignments of whole bacterial genomes. *IEEE Transactions on Visualization and Computer Graphics* 9, 3 (July-Sept. 2003), 361–377.
- [154] WRIGHT, W. Research report: information animation applications in the capital markets. In *Proceedings of the 1995 IEEE Symposium on Information Visualization* (1995), pp. 19–25.
- [155] XIAO, L., GERTH, J., AND HANRAHAN, P. Enhancing visual analysis of network traffic using a knowledge representation. In *VAST '06: Proceedings of the IEEE Symposium on Visual Analytics Science And Technology, 2006* (31 2006-Nov. 2 2006), pp. 107–114.
- [156] YEUNG, L. K., YAN, H., LIEW, A. W.-C., SZETO, L. K., YANG, M., AND KONG, R. Measuring correlation between microarray time-series data using dominant spectral component. In *APBC '04: Proceedings of the second conference on Asia-Pacific bioinformatics* (Darlinghurst, Australia, 2004), Australian Computer Society, Inc., pp. 309–314.
- [157] YI, J. S., KANG, Y. A., STASKO, J., AND JACKO, J. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (2007), 1224–1231.
- [158] YI, J. S., MELTON, R., STASKO, J., AND JACKO, J. A. Dust & magnet: multivariate information visualization using a magnet metaphor. *Information Visualization* 4, 4 (2005), 239–256.
- [159] ZELENY, M. Management support systems: Towards integrated knowledge management. *Human Systems Management* 7 (1987), 59–70.